

Hochschule Karlsruhe – Technik und Wirtschaft

Fakultät für Wirtschaftswissenschaften, Studiengang Technische Redaktion

Master-Thesis

**Konzeption und technische Realisierung eines Internetauftrittes
mit seinen redaktionellen Prozessen
für eine Softwareforschungsgruppe**
Schwerpunkte: Automatisierungsprozesse und standardisierte
Dokumentation von Softwareforschungsprojekten

Bearbeitungszeitraum: 01.06. – 22.10.2007

eingereicht von:
Fabienne Holdermann
Saldingerstraße 13a
76327 Pfinztal/Wöschbach

Tel.: 0175-2436729
E-Mail: Fabienne.Holdermann@web.de
Matr.-Nr.: 22828

Erstkorrektor: Prof. Martin Schober
Zweitkorrektor: Prof. Dr. Wolfgang Ziegler
Betreuer: Dipl.-Ing. Volker Hartmann, Forschungszentrum Karlsruhe

Danksagung

Für die Betreuung meiner Diplomarbeit möchte ich mich bei Herrn Prof. Schober und Herrn Prof. Ziegler herzlich bedanken.

Am Forschungszentrum Karlsruhe gilt mein besonderer Dank meinem Betreuer Herrn Hartmann für seine tatkräftige Unterstützung.

Herrn Dapp danke ich für seine Hilfsbereitschaft bei der Konfigurationsprogrammierung des *Authentic*. Bedanken möchte ich mich auch bei der gesamten Grid-Softwaregruppe, die mich durch ihre Teilnahme an der Umfrage, den Usability-Tests und durch nützliche Anregungen unterstützt hat.

Inhaltsverzeichnis

Abbildungsverzeichnis	V
Tabellenverzeichnis	VI
1 Einleitung	1
1.1 Forschungsprojekte und ihre Präsentation	1
1.2 Ziel der Master-Thesis	1
1.3 Aufbau der Master-Thesis	2
2 Ausgangssituation: Themenfindung	4
3 Grundlagen und Methoden	5
3.1 Einführung	5
3.2 Grid-Computing	5
3.3 Batch-Programmierung	5
3.4 Ist-Analyse	5
3.5 Bewertungsverfahren für Toolauswahl	6
3.6 Funktionsdesign	8
3.7 Modularisierung	9
3.8 Single Source Publishing und Cross Media Publishing	9
3.9 Informationsmodell	10
3.10 Metadaten	11
3.11 Usability und Überprüfungsmethoden	11
3.12 Redaktionsleitfaden	15
4 Themenfokussierung	17
4.1 Einführung	17
4.2 Vorgehensweise	17
4.3 Ergebnis: Schwachstellen und Erfüllungskriterien	21
4.4 Ergebnis: Thema der Master-Thesis	24
5 Technologie-Architektur	28
5.1 Einführung	28
5.2 Vorgehensweise bei der Definition der Technologie-Architektur	28
5.3 Ergebnis der Technologie-Architektur	30
6 Automatisierungsprozesse	33
6.1 Einführung	33
6.2 Vorgehensweise bei Konzeption und Realisierungsphase	33
6.3 Automatisierungskonzept und technische Realisierung	34
6.3.1 Automatisierungsprozess: Neues Projekt anlegen	35
6.3.2 Automatisierungsprozess: Links auf Glossartermdateien setzen	38
6.3.3 Automatisierungsprozess: Glossar generieren	40

6.3.4	Automatisierungsprozess: Formel im TeX-Format in Bild im PNG-Format umwandeln	41
6.3.5	Automatisierungsprozesse: Metadaten eintragen und Anschlussprozesse ausführen	42
6.3.6	Weitere Automatisierungsprozesse	44
7	Ablagekonzept und seine Realisierung	47
7.1	Einführung	47
7.2	Vorgehensweise bei der Konzeption	47
7.3	Ergebnis der Ordnerstruktur	48
7.4	Ergebnis der Metadatenregelungen für Dateien	49
8	Webauftrittskonzept	52
8.1	Einführung	52
8.2	Problematische Rahmenbedingungen	52
8.3	Webauftrittstruktur und Webseitenkonzept	53
8.3.1	Vorgehensweise bei der Konzeption	53
8.3.2	Ergebnis des Webauftritt- und Webseitenkonzeptes	56
8.3.2.1	Zielgruppen	56
8.3.2.2	Sprachen des Webauftrittes	56
8.3.2.3	Webauftrittstruktur	57
8.3.2.4	Webseitentypen	60
8.3.2.5	Metadaten der Webseitentypen	64
8.4	Linkkonzept	65
8.4.1	Vorgehensweise bei der Konzeption	65
8.4.2	Ergebnis des Linkkonzeptes	65
9	Technische Realisierung des Webauftrittskonzeptes	68
9.1	Einführung	68
9.2	Realisierung der Inhaltsstruktur mit eigenem Informationsmodell	68
9.3	Realisierung von XML-Templates	70
9.4	Modularisierung der XSLT-Stylesheets und Beispiele	71
9.5	Realisierung von automatisierten Webseiten	75
9.5.1	Automatisch eingefügte Inhalte	75
9.5.2	Automatisch generierte Webseite: Aktuelle Projekte	79
9.5.3	Automatisch generierte Webseiten: Startseite und Projektstartseite	80
9.6	Realisierungsbeispiele für Linktypen	81
9.6.1	Link auf Glossartermdatei	81
9.6.2	Link auf Seitenabschnitt	81
9.6.3	E-Mail Link mit verschlüsselter E-Mail Adresse	82
9.7	Realisierung eines zentralen Layouts	83

9.8	Beispielumsetzung von Webseitentypen mit SPS-Stylesheets	83
9.9	Realisierung einer <i>Authentic</i> -Projektdatei	84
10	Redaktionsleitfaden	85
11	Zusammenfassung und Ausblick	86
11.1	Zusammenfassung der Ergebnisse	86
11.2	Ausblick	88
	Literaturverzeichnis	90
	Anhang	96
A.	Fragenkatalog zur Ist-Analyse am Beispiel <i>Autor</i>	96
B.	Bewertungsmatrix und Präferenzmatrix am Beispiel <i>Redaktionelle Anforderungen</i>	103
C.	Bewertungsmatrix für Formel in Bild Umwandlung und eine bewertete Lösung	107
D.	Beispiel für Batch-Dateien	112
E.	Ordnerstruktur	119
F.	Alter und neuer Webaufttritt des IPE	121
G.	Beschreibungsmatrizen zu den Webseitentypen am Beispiel <i>Projektüberblicksseite</i>	124
H.	Grafische Darstellung der Linktypbeziehungen	136
I.	Beispiele für Complex Types	138
J.	Beschreibungsmatrizen der modularen Stylesheets an den Beispielen <i>metadaten_generate.xsl</i> und <i>email_link.xsl</i>	141
K.	Beispiele für XSLT-Stylesheets und JavaScript-Dateien	149

Abbildungsverzeichnis

Abbildung 1-1: Ziel der Master-Thesis	2
Abbildung 2-1: Ausgangssituation	4
Abbildung 3-1: Präferenzmatrix	7
Abbildung 3-2: Angebotsbewertung	7
Abbildung 3-3: Beispiel für Handlungssequenz nach dem Funktionsdesign	8
Abbildung 4-1: Einteilungsmöglichkeit für Softwareprojektdokumentation	19
Abbildung 4-2: Unterteilungsmöglichkeit der Systemdokumentation	20
Abbildung 4-3: Unterteilungsmöglichkeit der Benutzerdokumentation	20
Abbildung 4-4: Anzahl der Aussagen pro Schwachstellenkategorie	21
Abbildung 4-5: Erstellung einer Webseite mit Word-Editor	25
Abbildung 4-6: Webauftritt für die Projekte des IAI	27
Abbildung 4-7: Eine Projektbeschreibung bei IRS	27
Abbildung 5-1: Editoren der Technologie-Architektur	30
Abbildung 6-1: Beispiel für Codekommentierung in standardisierter Form	34
Abbildung 6-2: Templatenamen vs. angepasste Dateinamen für Projekt <i>GridIJ</i>	36
Abbildung 6-3: Prototyp-Projektordner vs. Projektordner für Projekt <i>GridIJ</i>	37
Abbildung 6-4: XML-Template mit Platzhaltern	37
Abbildung 6-5: XML-Template für Projekt <i>GridIJ</i>	38
Abbildung 6-6: Aufrufbeispiel für eine Batch-Datei mit Parameterübergabe	38
Abbildung 6-7: Prozessablauf <i>Links auf Glossartermdateien setzen</i>	39
Abbildung 6-8: Glossartermdatei mit allen acht Benennungsangaben	40
Abbildung 6-9: Prozessablauf <i>Glossar generieren</i>	40
Abbildung 6-10: TeX-Templatedatei zur Formelerstellung	42
Abbildung 7-1: Ordnerstruktur von <i>GRID_projects</i> und eines unbenannten Projektordners	48
Abbildung 7-2: Metadatenabschnitt <i>document_data</i> in einer XML-Webseitendatei	50
Abbildung 7-3: Befüllungsmethoden der Metadaten einer XML-Datei	50
Abbildung 8-1: Übersicht zum gesamten Webauftritt	57
Abbildung 8-2: Struktur eines Projekt-Webauftrettes	58
Abbildung 8-3: Projekttyp abhängige optionale und obligatorische Webseiten ..	59
Abbildung 8-4: Teaser der Startseite auf die Kontaktseite	66
Abbildung 9-1: Elementbenennung nach Funktionsdesign und Layout	69
Abbildung 9-2: Elemente <i>email</i> und <i>phone</i> mit eingeschränktem Datentyp <i>string</i>	70
Abbildung 9-3: Referenzierung der modularen Stylesheets im Stylesheet <i>overview_project.xsl</i>	71
Abbildung 9-4: Codeabschnitt der globalen Variablen im Stylesheet <i>index_index_project.xsl</i>	73
Abbildung 9-5: Einbindung der Navigation über PHP	75
Abbildung 9-6: Eintrag für Webseitentyp <i>Ziele</i> in der Textmoduldatei	76
Abbildung 9-7: Tabellenzelleninhalte dynamisch mit Textmodulen befüllen	77
Abbildung 9-8: Auslesen von Textmodulen in mehrfach verwendetem Template	78
Abbildung 9-9: Webseite <i>Aktuelle Projekte</i>	79
Abbildung 9-10: Webseite <i>Projektsartseite</i> für Projekt <i>GridIJ</i>	80
Abbildung 9-11: Setzen des Linkankers	81
Abbildung 9-12: SPP-Projekttemplatedatei für Webseiten eines Softwareprojektes	84

Abbildung 11-1: Ausschnitt aus der alten Startseite der Grid-Softwaregruppe...	87
Abbildung 11-2: Neue deutsche Startseite der Grid-Softwaregruppe.....	87
Abbildung 11-3: Neue englische Startseite der Grid-Softwaregruppe	88

Tabellenverzeichnis

Tabelle 4-1: Thema als Antwort auf die Anforderungen	25
Tabelle 5-1: Technologien und ihre Funktion im Redaktionsprozess	32
Tabelle 6-1: Übersicht zu den Automatisierungsprozessen	35
Tabelle 6-2: Prozessabläufe zum Eintragen von Metadaten	44
Tabelle 6-3: Ablauf weiterer Automatisierungsprozesse	46
Tabelle 7-1: Erweiterungsmöglichkeit der Ordnerstruktur	49
Tabelle 8-1: Gegensatz der Ausgangsbedingungen alter/neuer IPE-Webauftritt	52
Tabelle 8-2: Matrix zur Zielgruppenbeschreibung	56
Tabelle 8-3: Funktionen und Zielgruppen der Webseitentypen.....	62
Tabelle 8-4: Linktypen und ihre Funktion im Webauftritt	67

1 Einleitung

1.1 Forschungsprojekte und ihre Präsentation

Ob interne oder externe Dokumentation eines Forschungsprojektes - für eine Forschungsgruppe sind besonders vier Aspekte entscheidend:

- ihre wissenschaftlichen Publikationen als ein wichtiges Messkriterium ihrer Leistungsfähigkeit
- das rechtzeitige Veröffentlichen einer neuen Entwicklung, um nicht als „Zweiter“ den Urheberstatus für die Entwicklung zu verlieren
- die Präsenz und der Bekanntheitsgrad auf dem nationalen und internationalen Markt
- die weitere Beteiligung an künftigen Forschungsprojekten

Die Präsentation der Entwicklungen nach Innen und Außen spielt folglich eine relevante Rolle. Gerade das Internet stellt eine gute Möglichkeit dar, um neue Ergebnisse schnell weltweit zu verbreiten.

Auf der *German e-Science Conference 2007 (GES 2007)*¹ brachte Lagoze² in seinem Vortrag über *Knowledge Networking* die Notwendigkeit zur Sprache, in der Softwareforschung Wissen kompakt und strukturiert anderen zur Verfügung zu stellen.

1.2 Ziel der Master-Thesis

Ziel der Master-Thesis ist es, für die Grid-Softwaregruppe am Forschungszentrum Karlsruhe (FZK) eine Lösung zu finden, wie sie sich selbst und ihre Softwareforschungsprojekte in standardisierter Form schnell und einfach im Internet präsentieren kann:

¹ Die Konferenz fand in Baden-Baden vom 02.-05. Mai statt, auf der die Autorin einen Tag lang die Fragen der Teilnehmer am Stand der Grid-Softwaregruppe verfolgte.

² Carl Lagoze ist Forscher des *Information Science Program* an der Cornell Universität USA. Ziel des Programms: "Information Science brings together faculty, students and researchers who share an interest in combining computer science with the social sciences of how people and society interact with information." Quelle: <http://www.infosci.cornell.edu/> [Zugriff: 03.09.2007 09:10 MESZ]

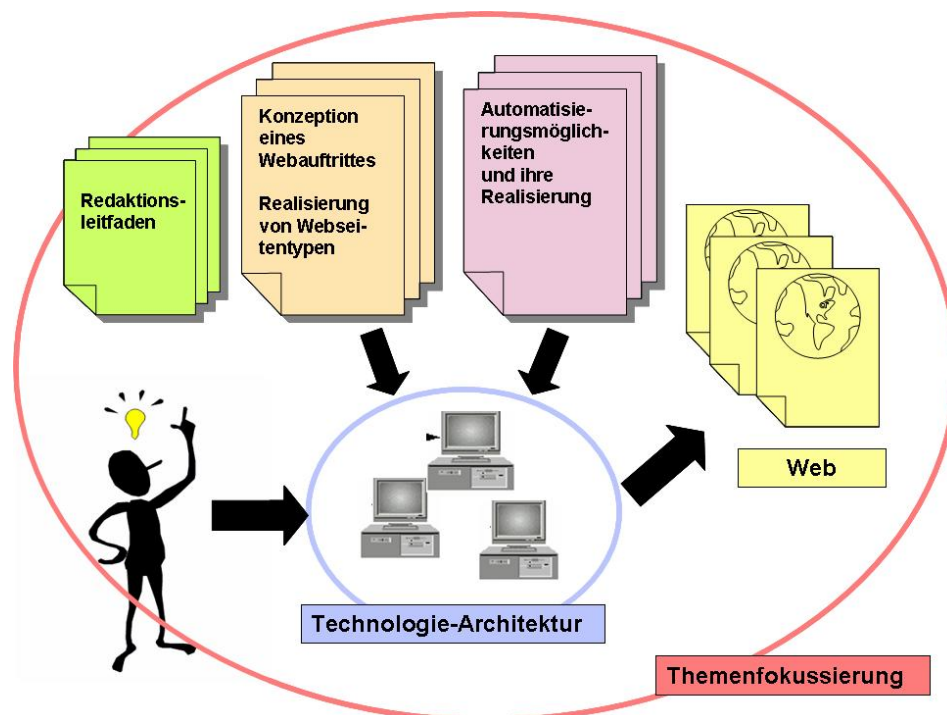


Abbildung 1-1: Ziel der Master-Thesis

Quelle: Eigene Darstellung

Die Master-Thesis umfasst im Detail folgende Teilziele:

- Themenfokussierung
- Definieren einer Technologie-Architektur für Online-Dokumentationsprozesse
- Aufzeigen von Automatisierungsmöglichkeiten und Realisierung ausgewählter Automatisierungsprozesse
- Konzeption eines Webauftrittes für die Grid-Softwaregruppe und für ihre Projekte
- Technische Realisierung von Webseitentypen
- Erstellung eines Redaktionsleitfadens

Bei der Konzeption und Umsetzung war der Aspekt zu berücksichtigen, dass es sich bei der Grid-Softwaregruppe nicht um Technische Redakteure, sondern um Softwareentwickler handelt. Entsprechend mussten zum Teil andere Maßstäbe gesetzt werden als in einer Technischen Redaktion üblich.

1.3 Aufbau der Master-Thesis

Die vorliegende Arbeit gliedert sich in elf Kapitel.

Im Anschluss an das einleitende Kapitel wird im zweiten die Ausgangssituation dargestellt, die beim Antritt der Master-Thesis vorgefunden wurde. Hierzu wird die Grid-Softwaregruppe und ihre Dokumentationssituation kurz vorgestellt.

Im dritten Kapitel werden grundlegende Begriffe und Methoden primär aus der Technischen Redaktion erläutert, die im praktischen Teil der Arbeit eine wichtige Rolle spielen.

Die Kapitel vier bis zehn umfassen den praktischen Teil der Master-Thesis. Es werden die einzelnen Ziele und ihre Umsetzung beschrieben:

In Kapitel vier wird die Themenfokussierung beleuchtet. Im nächsten Kapitel werden Vorgehensweise und Ergebnis der Technologie-Architektur aufgezeigt. Kapitel sechs stellt mit den Automatisierungsprozessen einen Schwerpunkt der Arbeit vor. Das Kernstück bilden die Umsetzungen einiger Prozesse z. B. mithilfe der Batch-Programmierung. In Kapitel sieben folgt die Darstellung des Ablagekonzeptes. Den nächsten Schwerpunkt der Arbeit bilden die Kapitel acht und neun. Kapitel acht stellt das Webaufttrittskonzept vor und geht dabei unter anderem auf die Webseitenstruktur, die Webseitentypen sowie das Linkkonzept ein. Besonderheiten werden aufgezeigt und Entscheidungen argumentativ begründet. Beim Kapitel neun zur technischen Realisierung des Webaufttrittskonzeptes werden Highlights herausgearbeitet. Der praktische Teil endet mit einem kurzen Kapitel zum Redaktionsleitfaden.

Das letzte Kapitel schließt mit einer Zusammenfassung und einem Ausblick über weitere Anwendungsszenarien der Softwaredokumentation.

2 Ausgangssituation: Themenfindung

Die Grid-Softwaregruppe am FZK entwickelt Softwarekomponenten und Anwendersoftware im Bereich der Grid-Technologien. Das Team besteht aus vier Angestellten und einer schwankenden Anzahl an stets wechselnden studentischen Hilfskräften und Auszubildenden aus dem Informatikbereich. Neben ihrer Haupttätigkeit – der Softwareentwicklung – sind sie für ihre Dokumentation selbst verantwortlich. Beim Antritt der Master-Thesis wurde folgende Situation vorgefunden:

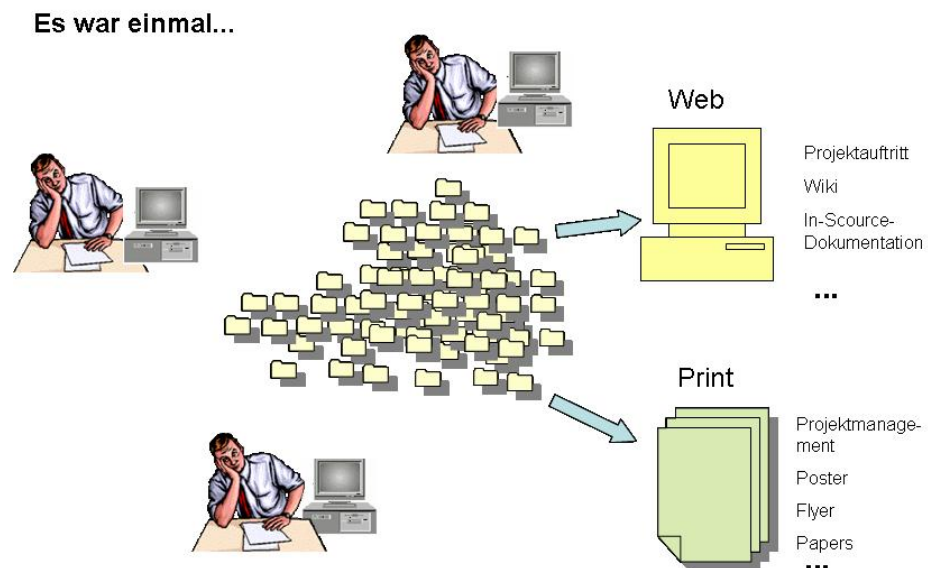


Abbildung 2-1: Ausgangssituation

Quelle: Eigene Darstellung

Die Dokumentation der Forschungsprojekte wird nebenher getätigt und eher stiefkindlich behandelt. Folglich existiert zu den einzelnen Softwareprojekten nur wenig Dokumentation, die auf Anweisung geschrieben wird.

Die Besonderheit der Master-Thesis besteht darin, dass es von Seiten der Grid-Softwaregruppe keine genauen Vorstellungen darüber gab, was und wie etwas an der gegebenen Dokumentationssituation verbessert werden kann. Für die Master-Thesis bedeutete dies, aus der Menge an Dokumentationsbrennpunkten ein Thema zu definieren, das so viele Schwachstellen im Dokumentationsprozess wie möglich beseitigt (siehe Kapitel 4.3).

3 Grundlagen und Methoden

3.1 Einführung

Zunächst werden zwei Begriffe aus der Informatik erklärt. Anschließend folgen Beschreibungen typischer Methoden aus der Technischen Redaktion, die angewandt wurden.

3.2 Grid-Computing

Die Anforderungen an die Leistungskapazität eines Computers steigen heutzutage rasant: Immer größere Datenmengen müssen gespeichert und möglichst zeitnah ausgewertet und dargestellt werden. Dies zeigt sich gerade in der Wissenschaft wie bei Großexperimenten der Teilchenphysik oder auch bei rechenintensiven Simulationen in der Industrie. Die Leistung der einzelnen Computer reicht in der Regel nicht mehr aus. Hier bietet sich Grid-Computing als Lösung an. Sobald Leistung benötigt wird, kann sie über ein Netz angefordert werden. Grid Computing stellt hierfür primär nicht genutzte IT-Ressourcen, die weltweit verteilt sein können, wie Rechenleistung, Speicherplatz, Anwendungen oder Netzwerke als „ganzheitliche Ressource“ zur Verfügung. IT-Betriebsmittel werden so leicht zugänglich und verfügbar gemacht wie Strom aus einer Steckdose.

3.3 Batch-Programmierung

Batch ist eine Script-Sprache, in der Stapeldateien – auch Batch-Dateien oder Scripts genannt – für MS-DOS geschrieben sind. Eine Batch-Datei enthält eine Abfolge von Befehlen des Betriebssystems, die beim Aufruf der Datei unter Kontrolle des Kommando-Interpreters des Betriebssystems nacheinander abgearbeitet werden (vgl. Voss 2000:779f). Die Stapelverarbeitung leistet z. B. unter Windows wertvolle Arbeit für die Umsetzung von Automatisierungsprozessen.

3.4 Ist-Analyse

In der Technischen Redaktion spielt die Ist-Analyse bei der Einführung redaktioneller Systeme (z.B. Content-Management-System) eine wichtige Rolle. Sie bildet den Ausgangspunkt, indem sie überprüft, ob Bedingungen und Anforderungen in einem Unternehmen für oder gegen eine Einführung sprechen. Alle redaktionellen Prozesse sowie Art, Umfang und Qualität der Informationsprodukte werden analysiert, um Schwachstellen und Effizienzpotentiale aufzudecken und die individuellen Anforderungen eines Unternehmens an eine Systemlösung zu identifizieren. Die Ergebnisse bilden das Fundament für die weitere Systemkonzeption und -planung (vgl. Fritz et al. 2005:68, Rockley et al. 2003:79f, siehe Kapitel 4.2).

3.5 Bewertungsverfahren für Toolauswahl

„There is no perfect solution. There are pros and cons for every tool and supporting technology. What works well for one organization may not work well for another. Your best defense is to be armed with much information as possible, to develop detailed evaluation criteria, and to test before you buy.“ (Rockley 2003:266).

Bewertungsmatrix

Hat ein Unternehmen der Beschaffung von redaktionellen Tools oder Systemen zugestimmt, gilt es, aus der Angebotslandschaft dasjenige auszuwählen, das die Bedürfnisse und Anforderungen der Redaktion bestmöglich erfüllt. Um die verschiedenen Alternativen vergleichend zu bewerten, eignet sich als Verfahren die Bewertungsmatrix. Das Verfahren umfasst folgende Punkte:

- Das Aufstellen von messbaren, eindeutig zu beantwortenden und positiv formulierten Bewertungskriterien, die im Idealfall gewichtet werden. Die Ergebnisse der Ist-Analyse sollten die Basis zu den spezifischen Anforderungen und Bedürfnissen der Redaktion bilden.
- Bewertung der Alternativen
- Ermittlung des Ergebnisses und Auswahl des Systems

(vgl. Fritz et al. 2005:81).

Es wird empfohlen, die Bewertungskriterien zu gruppieren und die Haupt- und untergeordneten Kriterien zu gewichten. So wird berücksichtigt, dass nicht alle Kriterien für die Auswahl dieselbe Bedeutung besitzen. Auch sollten Vertreter aller betroffenen Anwendergruppen, z. B. IT-Fachleute oder Redakteure, die Gewichtung durchführen, um die verschiedenen Sichtweisen mit einzubeziehen (vgl. Fritz et al. 2005:81ff).

Anwendung einer Präferenzmatrix

Als Hilfsmittel zur Gewichtung eignet sich eine Präferenzmatrix, wie die in Abbildung 3-1 dargestellte von Kirchhofer:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	
1	Präferenzmatrix													
2	für das Projekt				Projektbeispiel									
3	Gewicht	Rangfolge	Anzahl Nennungen	Kriterien										
4														
5														
6	a	20	2	2	a	Anbieterbezogene Kriterien								b
7														a
8		40	1	4	b	Lösungsbezogene Kriterien								b
9														d
10		10	4	1	c	Einsatz- und Reaktionszeit								b
11														a
12		10	4	1	d	Verträge								b
13														e
14		20	2	2	e	Kosten								e
15														
16		0	6	0	f									
17														
18		0	6	0	g									

Abbildung 3-1: Präferenzmatrix

Quelle: Beat Kirchhofer (2005)

Sowohl die Hauptkriterien zueinander werden bewertet, als auch die Unterkriterien von jeweils einem Hauptkriterium. Für jedes Kriterienpaar wird einzeln überprüft, welches Kriterium von den beiden wichtiger ist oder ob sie dieselbe Relevanz besitzen. Alle Gewichtungen zusammen müssen jeweils 100% ergeben (a) (vgl. Kirchhofer 2005).

Vergleichende Angebotsbewertung

Für die Bewertung der Angebote anhand der gewichteten Bewertungskriterien eignet sich das Tool *Angebotsbewertung* von Kirchhofer.

A	B	C	D	E	F	G	H	I	J	K	L	M	N
Angebotsauswertung							Ebene 1	Ebene2	Wert	Angebot Fa. 1		Angebot Fa. 2	
1. Kriteriencluster							X%			Note 1 - 6	Wert	Note 1 - 6	Wert
1.1							X%	#N/A	0	#N/A	0	#N/A	
1.2							X%	#N/A	0	#N/A	0	#N/A	
1.3							X%	#N/A	0	#N/A	0	#N/A	
1.4							X%	#N/A	0	#N/A	0	#N/A	
1.5							X%	#N/A	0	#N/A	0	#N/A	
2. Kriteriencluster							X%						
2.1							X%	#N/A	0	#N/A	0	#N/A	
2.2							X%	#N/A	0	#N/A	0	#N/A	
2.3							X%	#N/A	0	#N/A	0	#N/A	
2.4							X%	#N/A	0	#N/A	0	#N/A	
2.5							X%	#N/A	0	#N/A	0	#N/A	
2.6							X%	#N/A	0	#N/A	0	#N/A	
2.7							X%	#N/A	0	#N/A	0	#N/A	
2.8							X%	#N/A	0	#N/A	0	#N/A	
2.9							X%	#N/A	0	#N/A	0	#N/A	
2.10							X%	#N/A	0	#N/A	0	#N/A	
2.11							X%	#N/A	0	#N/A	0	#N/A	
2.12							X%	#N/A	0	#N/A	0	#N/A	
2.13							X%	#N/A	0	#N/A	0	#N/A	
Total							0%	0%					

Abbildung 3-2: Angebotsbewertung

Ouelle: Beat Kirchhofer (2005)

Hier werden die Gewichtungen der Hauptkriterien (a) und der untergeordneten (b) eingetragen. Den Produktangeboten werden pro Bewertungskriterium eins bis sechs Punkte gegeben (c). Ein Punkt bedeutet, dass das Kriterium nicht und sechs Punkte, dass es optimal erfüllt ist. Das Angebot mit der höchsten Punkt-

zahl (d) gilt als Favorit. Es bietet das beste Preis-/Leistungsverhältnis (vgl. Kirchhofer 2005).

3.6 Funktionsdesign

Definition des Funktionsdesigns

„Das Funktionsdesign ist eine universelle und flexible Standardisierungs-, Strukturierungs- und Schreibtechnik [...] [und] eignet sich besonders gut, die Planung und Erstellung Technischer Dokumentation zu unterstützen.“ (Muthig/Armbruster-Schäferlein 1999:1995).

Bei der von Muthig und Armbruster entwickelten Technik werden für jede Dokumentart Elemente festgelegt, die für den kommunikativen Zweck erforderlich sind:

- Sequenzmuster (z.B. eine Handlungssequenz (a))
- Funktionale Einheiten (z.B. Handlungsaufforderung (b), Handlungsergebnis (c))
- Auszeichnungselemente (z.B. Auszeichnung einer Taste (d))

Beispiel: Handlungssequenz

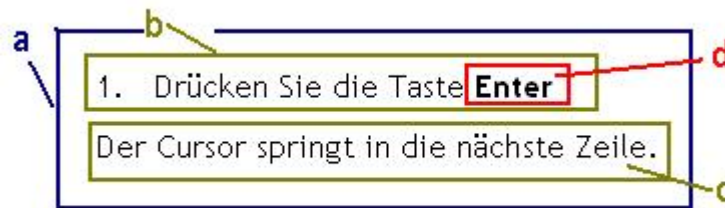


Abbildung 3-3: Beispiel für Handlungssequenz nach dem Funktionsdesign

Quelle: Eigene Darstellung

Ein Dokument setzt sich aus Sequenzen zusammen. Diese werden wiederum aus Funktionalen Einheiten gebildet, wobei eine Funktionale Einheit aus einem Satz oder auch aus mehreren Sätzen bestehen kann. In einer Funktionalen Einheit können Auszeichnungselemente vorkommen.

Zudem müssen für jedes Element Regeln definiert werden, die der Autor bei deren Verwendung einhalten muss und die im Idealfall in einem Redaktionsleitfaden festgehalten sind, um eine konsistente Dokumentationserstellung durch die Autoren zu gewährleisten. Diese Regeln betreffen vier Kategorien:

- *Verwendung*
- *Sequenzierung*
- *Innere Struktur und syntaktisches Muster bzw. Formulierungsmuster*
- *Layout*

(vgl. Muthig/Armbruster 1999:1995ff).

Funktionsdesign und XML	Die Einhaltung der Regeln wird innerhalb der Autorenwerkzeuge typischerweise durch Formatvorlagen oder XML-Elemente unterstützt. Gerade für XML-basierte Dokumentation ist das Funktionsdesign geeignet, da die Sequenzierungsregeln und hierarchischen Strukturen 1:1 über eine DTD oder ein Schema abgebildet werden können.
-------------------------	--

Nutzen des Funktionsdesigns	Das Funktionsdesign hilft, die Dokumentation konsistent, effizient und anwenderfreundlich zu gestalten. Die Standardisierung und Konsistenz der Dokumentation bildet wiederum die Basis für die Wiederverwendung von Inhalten und Automatisierung der Dokumentationserstellung (siehe 9.2).
-----------------------------	---

3.7 Modularisierung

Definition von Modularisierung	Modularisierung bedeutet, Informationsbausteine – auch Module genannt (z. B. Text, Grafik) – festzulegen, um einen Pool an Informationsbausteinen bereitzustellen, aus dem sich der Inhalt von verschiedenen Dokumentationstypen für verschiedene Medien zusammensetzen lässt (siehe Kapitel 3.8). Ziel ist es, Redundanzen in Text und Grafik durch die Wiederverwendung (Reuse) der Module zu vermeiden. Zur Bestimmung der Module stehen verschiedene Methoden wie z. B. das Funktionsdesign (siehe Kapitel 3.6) zur Verfügung. Die Größe eines Moduls kann von Daten, über Sätze, Abschnitte und Kapitel bis hin zu Seiten und sogar ganzen Dokumenten reichen. Sie spielt eine wichtige Rolle, da sie bestimmt, inwieweit sich die Module wieder verwenden lassen (vgl. Ziegler 2007:52ff).
--------------------------------	--

Voraussetzungen für Wiederverwendung der Module	Modularisierung geht einher mit Standardisierung. Um Module wieder verwenden zu können, muss eine neutrale Datenerfassung (z. B. XML) und eine Klassifikation des Moduls und seines Inhalts vorhanden sein. Die Klassifikation betrifft die Größe, Struktur bzw. Sequenzierung, Metadaten und Benennung des Moduls sowie die Reuse-Methode (vgl. Ziegler 2007:52ff).
---	--

Vorteile der Modularisierung	Die Wiederverwendung von Modulen und ihre Kombinierbarkeit bewirken eine Erhöhung der Dokumentationsqualität sowie eine Reduzierung des Zeit- und Kostenaufwandes für die Dokumentationserstellung.
------------------------------	---

3.8 Single Source Publishing und Cross Media Publishing

Definition von SSP	Single Source Publishing (SSP) bedeutet, Informationen aus einer Quelle mehrfach wieder zu verwenden, primär für externe Dokumentation (z. B. Kundendokumentation wie Bedienungsanleitung, Schulungsunterlagen), aber auch für interne Dokumentation (z. B. Entwicklungs-, Fertigungsunterlagen).
--------------------	---

Das heißt, wie Closs zusammenfasst (vgl. Closs 2007:28):

- „denselben Inhalt nur einmal erstellen
- vorhandene Inhalte soweit wie möglich wieder verwenden
- aus einem Inhaltsbestand Ausgabedokumente produzieren, die sich in Umfang, Zusammenstellung und Darstellung unterscheiden
- denselben Inhalt in unterschiedlichen Layouts darstellen“.

Definition von CMP

Cross Media Publishing (CMP) betont hingegen, erstellte Dokumente in verschiedenen Zielformaten (z. B. HTML, PDF) und Medien (z. B. Papier, Internet, CD-Rom) zu nutzen. „Das Ziel ist die parallele Erzeugung beispielsweise von Print-, Web- oder Hilfe-Formaten.“ (Ziegler 2004:19). Cross Media Publishing schließt sich folglich an das Single Source Publishing an.

Voraussetzungen für SSP und CMP

Single Source Publishing und Cross Media Publishing setzen voraus:

- eine neutrale Datenerfassung bzw. die Trennung von Inhalt und Layout, z. B. über die Kombination von XML und XSL
- eine Zusammensetzung der Dokumentation aus klassifizierten Modulen
- die technische Unterstützung durch geeignete Tools

(vgl. Closs 2007:28, Ziegler 2007:37ff).

3.9 Informationsmodell

Definition des Informationsmodells

Mithilfe eines Informationsmodells wird unter anderem die Informationsstruktur eines Informationsproduktes³ bestimmt (vgl. Fritz et al. 2005:93). Das Informationsmodell verwendet zur Strukturierung eine bestimmte Methode (z. B. funktionalorientierte wie das Funktionsdesign oder layoutorientierte) und bildet die Struktur elektronisch ab: z. B. in Form eines Schemas, einer DTD oder einer DOT. Handelt es sich um modularisierte Dokumentation, bildet das Informationsmodell deren Struktur und Sequenzierung ab und dient damit deren konsistenter Einhaltung.

Bei den Informationsmodellen existieren Industrie-Standards wie *DocBook* oder *DITA*. Des Weiteren gibt es Informationsmodelle, die für die firmeninternen Informationsprodukte eigens erstellt oder von Tools- bzw. Systemanbietern vorgegeben wurden (vgl. Ziegler 2007:90ff).

³ Ein Informationsprodukt ist „any output consisting of content, such as user guides, catalogs, documents, brochures, press releases, annual reports, intranet, web sites, or technical specifications“ (Rockley 2003:160f).

DITA und DocBook	Das Informationsmodell <i>DocBook</i> wird von OASIS gepflegt und eignet sich besonders für das Verlagswesen oder die Dokumentation von Soft- und Hardware. <i>DITA</i> (Darwin Information Typing Architecture) von IBM/OASIS eignet sich besonders für Publikationen in Online-Medien und Help-Systemen (Rockley 2003:178f).
Vorteile des Informationsmodells	„An effective information architecture helps people (both internal and external) to your organization) find, manage, and continue to add to the existing information consistently, following defined rules for structure and semantics“. (Rockley 2003:161). Zudem trägt ein Informationsmodell z. B. durch konsistente Strukturierung der Informationsprodukte dazu bei, ihre Benutzerfreundlichkeit zu verbessern.

3.10 Metadaten

Für Metadaten finden sich in der Literatur verschiedene Definitionen. Am häufigsten fällt die Aussage, dass Metadaten *Daten über Daten* seien (vgl. Kammerer/Sieber 2006:56). Diese Aussage übersieht aber den elementaren Aspekt, dass „metadata can be used to describe the data's behavior, processes, rules, and structure.“(Rockley 2003:184).

Rockley gibt eine Übersicht, welche Möglichkeiten Metadaten bieten (vgl. Rockley 2003:184):

- „Effective retrieval
- Systematic reuse
- Automatic routing based on workflow status
- Tracking of status
- Reporting“.

Zu einem Dokument können beispielsweise Metadaten angegeben werden, die Auskunft darüber geben, wann und von wem das Dokument erstellt wurde, welches Produkt es beschreibt, ob es bereits publiziert ist, welche Zielgruppen es hat, für welche Medien es bestimmt ist oder in welchen Sprachen es erscheint.

Metadaten können entweder direkt im Dokument (z. B. XML-, HTML-Datei) angegeben werden oder in einer separaten Datei, die auf das Dokument referenziert.

3.11 Usability und Überprüfungsmethoden

Definition von Usability	Für die Disziplin <i>Usability</i> finden sich in Fachkreisen die verschiedensten Benennungen: <i>Usability Engineering</i> , <i>Usability Testing</i> oder <i>User Centered Design</i> (vgl.
--------------------------	---

Heinsen/Vogt 2003:2).⁴ Allgemein lässt sich *Usability* mit *Benutzerfreundlichkeit*, *Benutzbarkeit* oder mit *Gebrauchstauglichkeit* übersetzen (vgl. Hermann 2006, Schober 2004:10). Als weithin akzeptierte Definition gilt die von ISO 9241-11, nach der Usability „Effektivität, Effizienz und das Ausmaß der Zufriedenheit [ist], mit denen bestimmte Benutzer Ziele in vorgegebenen Umgebungen erreichen.“⁵ Das bedeutet: Ein Produkt, eine Software oder ein Webaufttritt sind benutzerfreundlich, wenn der Benutzer sein Ziel vollständig erreicht, der Aufwand zur Zielerreichung möglichst minimal und der Benutzer mit der eigenen Zielerreichung zufrieden ist, das Produkt mag. Usability bildet somit ein wichtiges Qualitätsmerkmal.

Für die Usability von Webauftritten lassen sich zahlreiche Empfehlungen und Richtlinien konstatieren, ob nun die ISO-Norm 9241-110 mit ihren *Grundsätzen der Dialoggestaltung* oder Teile der ISO-Norm 14915 zur *Software-Ergonomie für Multimedia Benutzungsschnittstellen* (vgl. Hermann 2006).

Regelauswahl für benutzerfreundlichen Webaufttritt

Die folgende Auflistung zeigt eine Auswahl nützlicher Regeln für einen Webaufttritt, der Usability-Anforderungen (siehe Kapitel 8) berücksichtigt (vgl. Bartel (2004:42ff.), Hoffmann et. al (2002:295ff., 386), Manhartsberger / Musil (2002: 22ff.), Schober (2004:16-19), Schulz (2006), Wirth (2005), Zimmermann (2001:17).

Navigation

Die Navigation wird benutzerfreundlich gestaltet durch ...

- kurze Navigationswege (drei Klicks bis zum Ziel)
- Basisnavigation, die auf allen Webseiten jederzeit sichtbar ist
- verständlich benannte Navigationsmenüpunkte
- eindeutig benannte Links. Das Linkziel muss erkennbar sein.
- Tooltips bei Grafiken, die als Links fungieren (falls Grafiken nicht angezeigt werden)
- konsistent verwendete Linktypen
- funktionierende Links
- Firmenlogo, das als Link zur Startseite dient

⁴In der Master-Thesis wird die Bezeichnung Usability verwendet. Grund: Um die Disziplin von der Methode Usability Testing klar abzugrenzen, mit der die Gebrauchstauglichkeit eines Produktes überprüft wird, wird diese Bezeichnung nicht benutzt. Usability Engineering und User Centered Design sind hinsichtlich ihrer semantischen Begriffsmerkmale zu eng gefasst und damit keine geeignete Bezeichnung für einen Oberbegriff.

⁵Qualitätsaspekte wie „joy of use“ bzw. die Bedeutung von Emotionen im Umgang mit interaktiven Produkten gewinnen in der Usability Szene immer mehr an Bedeutung. Beispielhaft sei der Ansatz von Hassenzahl et al. aufgeführt, die Usability als *Quality of use* und *Quality of Experience* ansehen. Diese Aspekte sind bereits im Kriterium Zufriedenheit angelegt (vgl. Heinsen/Vogt 2003:3, Hermann 2006, Hassenzahl et. al 2000:201ff.)

	▪ Suchmöglichkeiten
Seitenaufbau/ Bildschirmgröße	Der Seitenaufbau und die Bildschirmgröße werden benutzerfreundlich gestaltet durch ... ▪ Überschriften und Unterüberschriften zur Themengliederung ▪ kurze informative Einleitung, was den User auf der Webseite erwartet ▪ kurze Textabschnitte ▪ Berücksichtigung der verschiedenen Bildschirmauflösungen (600x800, 1024x768 etc.) ▪ leserfreundliche Satzspiegelbreite, die 700px nicht überschreitet ▪ Zeilenlänge, die 60 Zeichen nicht überschreitet ▪ kurze Webseiten, bei denen der User nicht mehr als zweimal scrollen muss
Bilder	Bilder werden benutzerfreundlich gestaltet durch ... ▪ kurze Ladezeiten ▪ Alternativtexte als Ersatz für nicht geladene Bilder ▪ ein Logo auf allen Webseiten ▪ eine Bildauflösung von 72ppi, da Standardbildschirme nur diesen Wert darstellen
Ladezeiten	Kurze Ladezeiten werden erreicht durch ... ▪ eine Dateigröße der Webseite inklusive Bilder und Skripte von nicht mehr als 100 KB⁶ ▪ Verzicht auf unnötige Multimedia-Effekte ▪ mehrere kleinere Layout-Tabellen anstelle einer großen oder den völligen Verzicht auf Layout-Tabellen
Inhalt	Der Inhalt wird benutzerfreundlich gestaltet durch ... ▪ aktuelle Informationen ▪ ein transparent gestaltetes Angebot ▪ einen schnell und klar ersichtlichen Inhalt und Nutzen des Webauftrittes und der Webseiten ▪ keine unfertigen Webseiten

⁶Die Empfehlungen in der Literatur zur Dateigröße schwanken: Bartel gibt z.B. 68 KB an (vgl. Bartel 2004:44)

- Berücksichtigung der Bedürfnisse der unterschiedlichen Zielgruppen
- Links auf weiterführende Webseiten für interessierte User
- Berücksichtigung des Vorwissens der Zielgruppen durch Erläuterung spezifischer Terminologie
- eine Webseite zu den Ansprechpartnern, die mindestens die E-Mail Adressen enthält, damit der User Kontakt aufnehmen kann
- Verzicht auf Pop-up-Fenster

Layout

Das Layout wird benutzerfreundlich gestaltet durch ...

- Einhaltung der CI-Richtlinien auf allen Webseiten
- Unterstreichungen nur für Linkauszeichnung
- Farbänderung der Links, wenn sie besucht wurden
- eine gut lesbare und auf allen Browsern verfügbare Schriftart
- eine gut lesbare Schriftgröße am Bildschirm
- Leerräume und Absätze für Sinneinheiten
- linksbündige Textausrichtung

Browser

Der Browser wird benutzerfreundlich gestaltet durch ...

- korrekte Anzeige bzw. volle Funktionsfähigkeit der Webseiten in allen gängigen Browsern
- korrekten Standard HTML Code

Seitenidentifizierung im WWW

Die Seitenidentifizierung im WWW wird erreicht durch ...

- eine aussagekräftige Titelzeile im Browserfenster
- Einsatz von Meta-Tags zum Auffinden der Webseiten über Suchmaschinen

Usability-Methoden

Um die Usability eines Produktes zu überprüfen, stehen verschiedene Methoden zur Auswahl (vgl. Hermann 2006, Heinsen/Vogt 2003:98ff). Beispielfhaft seien aufgeführt:

- *Wording-Tests* in Form von
 - *Begriffserklärungen*: Der Testperson wird ein Begriff vorgegeben, den sie erklären soll. Ziel ist zu überprüfen, ob er verstanden wird und welche Synonyme verwendet werden (siehe Kapitel 7.2).
 - *Card-Sorting*: Der Testperson werden Begriffe vorgegeben, die sie sortieren soll. Erfolgte die Methode ursprünglich mithilfe von Karten, die die Testperson gruppiert oder einordnet, wird sie heute auch per

Computer durchgeführt (vgl. Vogt 2003:238f). Sie gibt Antwort darauf, ob Informationsstruktur und Kategorien verstanden werden. Auch schwer kategorisierbare Begriffe lassen sich damit einordnen (siehe Kapitel 7.2).

- Usability Testing: Die Testperson bearbeitet „in einer möglichst realen Situation vorgegebene typische Aufgaben des Produktes aus ihrem Tätigkeitsfeld“ (Hennig 1998:248). Dabei wird sie beobachtet und aufgezeichnet. Die gewonnenen Daten werden analysiert und Optimierungsvorschläge erarbeitet. Ziel ist somit, die Benutzerfreundlichkeit eines Produktes praxisnah zu ermitteln (vgl. Hennig 1998:248).

Integration von Usability in den Entwicklungsprozess

Es genügt jedoch nicht, erst das fertige Produkt auf seine Benutzerfreundlichkeit hin zu überprüfen, denn Änderungen in dieser Phase sind aufwändig und teuer. Vielmehr müssen Usability-Maßnahmen in den Entwicklungsprozess integriert werden, um Anpassungen entsprechend der Ergebnisse aus den Testverfahren vorzunehmen und eine optimale Benutzerfreundlichkeit zu erzielen.⁷

3.12 Redaktionsleitfaden

Definition von Redaktionsleitfaden

Der Redaktionsleitfaden, auch Redaktionshandbuch⁸ genannt, ist im Idealfall eine „Bedienungsanleitung“ mit allen erforderlichen Informationen, um Informationsprodukte effizient und einheitlich in einer Redaktion zu erstellen. Er richtet sich in der Regel an alle Beteiligte eines Dokumentationsprozesses: Technische Redakteure, Entwickler und Konstrukteure, Illustratoren und Grafiker, Übersetzer, Zulieferer von Dokumentationsleistungen, Druckereien etc. (Oehmig 2004:7,36).

Inhalt eines Redaktionsleitfadens

Um den Zielgruppen Antwort auf all ihre Fragen zum Dokumentationsprozess zu geben und standardisierte, qualitativ hochwertige Dokumentation zu sichern, muss der Redaktionsleitfaden unter anderem Folgendes leisten:

- Alle redaktionellen Prozesse und die dazugehörigen Redaktionswerkzeuge definieren, um eindeutige und transparente Abläufe zu erhalten. Dazu gehören Planung, Informationsbeschaffung, Erstellung, Korrektur, Freigabe, Qualitätssicherung, Produktion, Verteilung, Wartung/Revision und Archivierung.

⁷Die Norm DIN EN ISO 13407 schreibt einen auf Benutzerfreundlichkeit hin ausgerichteten Prozess mit folgenden beiden Grundprinzipien vor: Eine aktive Beteiligung von echten Benutzern sowie ein mehrmaliges Durchlaufen der Gestaltungslösungen (vgl. DIN EN ISO 13407 2000:4)

⁸ In der Unternehmenswelt finden sich weitere Benennungen wie Style Guides, Dokumentationsrichtlinien oder Corporate Design-Handbücher, wobei jedes Unternehmen unterschiedliche Vorstellungen mit dem Begriff verbindet (vgl. Schwermer 1998).

- Arbeitsanweisungen liefern und Zuständigkeiten sowie Verantwortlichkeiten festlegen, damit jeder Beteiligte weiß, was er zu tun hat und wer Ansprechpartner ist.
- Alle Dokumentationsarten definieren, um einheitliche Informationsprodukte zu erhalten. Dies setzt Angaben voraus ...
 - zu den Zielgruppen
 - zur strukturellen und funktionalen Standardisierung des Inhalts inklusive angewandtem Informationsmodell (siehe Kapitel 3.9)
 - zur Gestaltung (Typografie, Layout)
 - zur Sprache (Terminologie- und Formulierungsrichtlinien)

(vgl. Fritz et al. 2005:96, Hoffmann 2002:40f, Oehmig 2004:9f, Schwermer 1998).

Nutzen eines Redaktionsleitfadens

Ein Redaktionsleitfaden bewirkt für eine Technische Redaktion ...

- eine Effizienzsteigerung bezüglich redaktioneller Abläufe und interner Zusammenarbeit
- kürzere Konzeptionsphasen für neue Dokumentation
- eine Vereinheitlichung von Inhalten und Gestaltung der Dokumentation
- Wissensspeicherung bezüglich der redaktionellen Prozesse
- schnelleres Einbinden von neuen Mitarbeitern und externen Dienstleistern

(vgl. Oehmig 2004:9f, Schwermer 1998, Fritz et al. 2005:96).

4 Themenfokussierung

4.1 Einführung

Die folgenden Kapitel zeigen auf, warum ein Lösungspaket für eine schnelle und einfache Online-Präsentation von Softwareforschungsprojekten in standardisierter Form viele Dokumentationsschwachstellen beseitigt und deshalb als Thema der Master-Thesis definiert wurde.

4.2 Vorgehensweise

Ist-Analyse:
Fragenkatalog

Ziel war es, im Rahmen einer Ist-Analyse die vorhandene Dokumentationssituation zu durchleuchten, aber auch einen Überblick zu den Projekten und ihren Charakteristika zu erhalten. Untersucht wurden:

- die allgemeinen Rahmenbedingungen der Lösung
- die Erwartungen und Anforderungen an eine Lösung
- die Dokumentationstypen und Medien
- die Zielgruppen der Dokumentation
- die eingesetzten Technologien
- die redaktionellen Abläufe
- die Schwachstellen im Dokumentationsprozess und in der Dokumentation selbst
- die Softwareprojekttypen und ihre Charakteristika

Die Erwartungen und Anforderungen an eine Lösung und die Schwachstellen sind zwei unterschiedliche Blickrichtungen auf denselben Sachverhalt, wobei die erste noch mehr Informationen beisteuern kann.

Eingesetzt wurde ein Fragenkatalog zu typischen Rollen, die ein Mitarbeiter bzw. Student in den redaktionellen Prozessen gemäß seiner Aufgaben einnehmen kann: Autor, Lektor, Übersetzer, Layouter, Freigeber, Herausgeber. Er berücksichtigt aber auch die Rolle des Fachgruppenleiters, um einen ersten Gesamtüberblick von der Dokumentationssituation zu erhalten sowie die Rolle des Softwareentwicklers, um die Projektcharakteristika und speziellen Anforderungen an die Dokumentation herauszufiltern. Auch Fragen an neue Studenten zu benötigten Projektinformationen und ihren Eindrücken hinsichtlich der Dokumentation sind enthalten. Für eine schnelle Klassifizierung der Antworten wurden zu den einzelnen Fragen die damit verbundenen Intentionen notiert. Die Perspektive des Dokumentationsnutzers floss ebenfalls mit ein: Zum einen durch das Feedback von Außen, das der Fachgruppenleiter schilderte, zum anderen durch die Soft-

wareentwickler selbst, da sie gleichzeitig auch Nutzer ihrer Dokumentation sind (siehe Anhang A).

Interviews

Zunächst wurde die Befragung nur mit einem Mitarbeiter durchgeführt, um etwaige Verbesserungen am Fragenkatalog vorzunehmen. Insgesamt wurden sieben Personen stellvertretend für die zum Zeitpunkt zehnköpfige Grid-Softwaregruppe befragt – Gruppenleiter, Mitarbeiter sowie Studenten – um verschiedene Sichten auf die Dokumentationssituation zu erhalten. Zudem ließ sich die Autorin von den Befragten Dokumentationsbeispiele der Grid-Softwaregruppe zeigen und suchte selbst nach benötigten Informationen, um Schwachstellen aufdecken und um die Aussagen besser nachvollziehen zu können.

Der Aspekt, dass manche Fragen auf dieselbe Antwort abzielen, aber aus unterschiedlichen Perspektiven gestellt werden, war wichtig, da sich bereits in der „Vorrunde“ gezeigt hatte, dass die Befragten je nach Rolle neue Aussagen zum selben Sachverhalt angaben. Nach Schwachstellen oder nach Wünschen an eine Lösung wurde direkt gefragt oder konnte an Aussagen zu Fragen wie – *Was sind die häufigsten Fehler, die redigiert werden* – abgelesen werden.

Dokumentationstypen für Softwareentwicklungsprojekte

Einen weiteren Anhaltspunkt für die Themenfindung boten die für ein Softwareentwicklungsprojekt notwendigen Dokumentationstypen. In Anlehnung an die *Richtlinien für die Softwaredokumentation* (2005:6ff), die sich an mehreren ISO/IEC-Normen⁹ orientieren, und angeregt vom Aufgaben bezogenen Einteilungsansatz für Softwaredokumentation von Geiger (1985:12), wurden folgende Dokumentationstypen zusammengestellt. Jeder Dokumentationstyp wird zur Verdeutlichung durch eine kurze Definition und Beispiele erläutert:

⁹ ISO/IEC 6592: 2000 *Information Technology – Guidelines for the Documentation of Computer-based Application Systems*, ISO/IEC FCD 9127:2005 *Software Engineering – User Documentation and Cover Information for Consumer Software Packages* und ISO/IEC 18019: 2004 *Software and System Engineering – Guidelines for the Design and Preparation of User Documentation for Application Software*

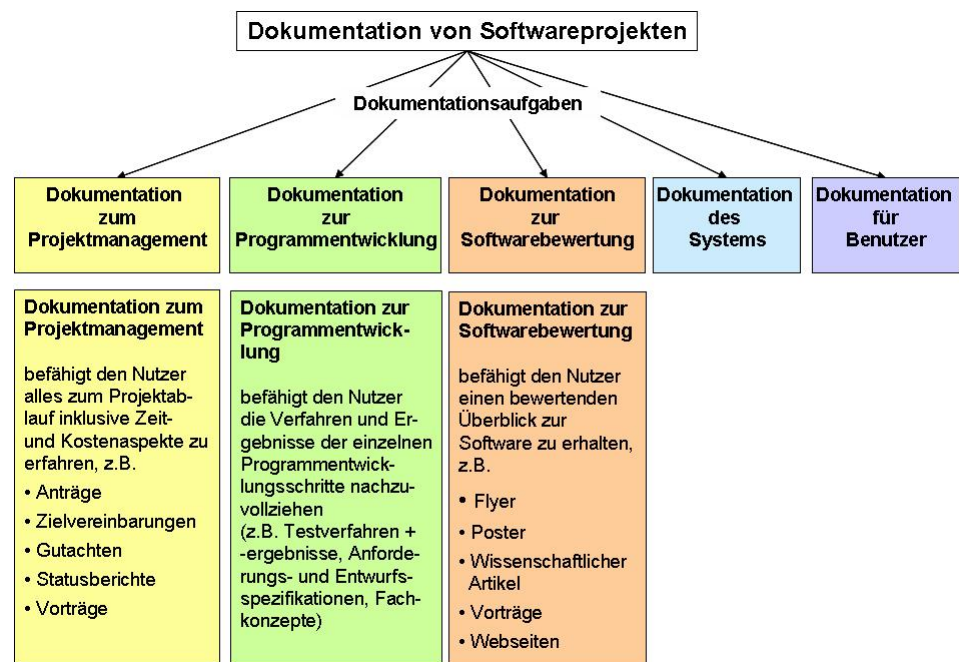


Abbildung 4-1: Einteilungsmöglichkeit für Softwareprojektdokumentation

Quelle: In Anlehnung an *Richtlinien für die Softwaredokumentation* (2005:6) und Geiger (1985:12)

In einem ersten Schritt lässt sich die Dokumentation nach ihrer zentralen Aufgabe einteilen: Dokumentation zum Projektmanagement, Programmentwicklung, Softwarebewertung, System- und Benutzerdokumentation.

Die Systemdokumentation kann hinsichtlich ihres Nutzungszwecks in zwei weitere Dokumentationstypen gegliedert werden:

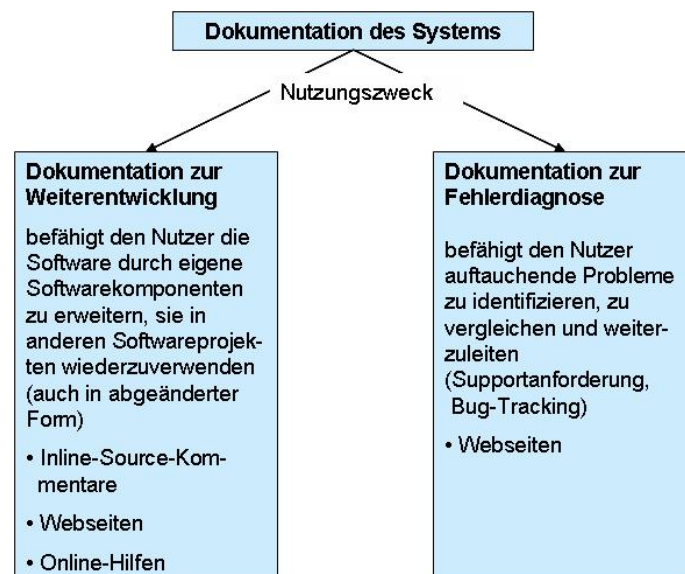


Abbildung 4-2: Unterteilungsmöglichkeit der Systemdokumentation

Quelle: In Anlehnung an Geiger (1985:12f)

Die Benutzerdokumentation lässt sich in fünf Kategorien unterteilen:

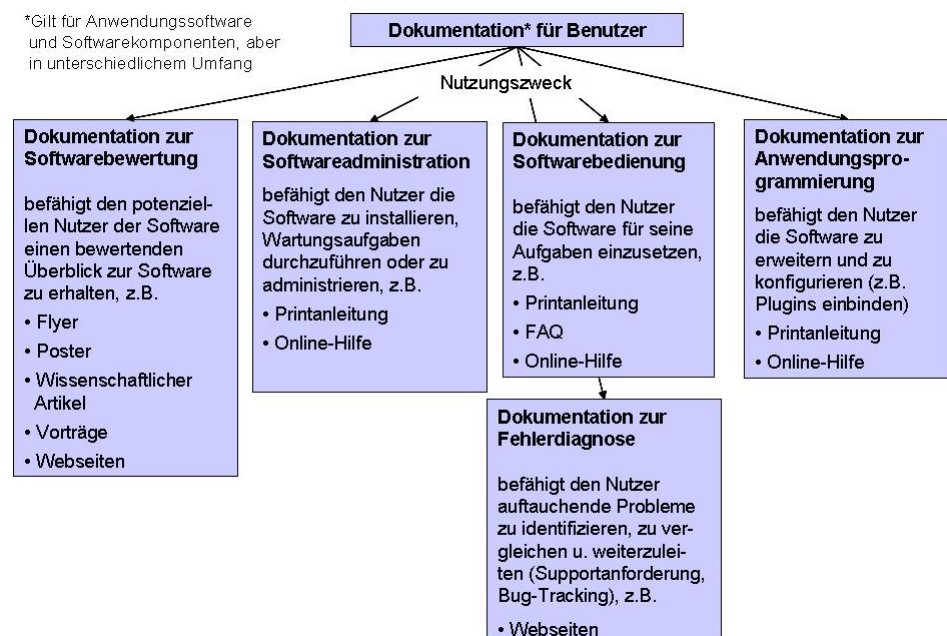


Abbildung 4-3: Unterteilungsmöglichkeit der Benutzerdokumentation

Quelle: In Anlehnung an Geiger (1985:12f)

Der Typ *Dokumentation zur Softwarebewertung* lässt sich sowohl der *Dokumentation von Softwareprojekten* als auch der *Dokumentation für Benutzer* zuordnen. Ebenso verhält es sich mit der *Dokumentation zur Fehlerdiagnose*. Zum einen

handelt es sich um eine Benutzerdokumentation, zum anderen dokumentiert sie auch das System bzw. seine Fehler (siehe Abbildung 4-2).

4.3 Ergebnis: Schwachstellen und Erfüllungskriterien

Für die Themenfindung waren neben den Verbesserungswünschen und Dokumentationsanforderungen auch die Schwachstellen der aktuellen Dokumentationssituation und die empfohlenen Dokumentationstypen maßgebend.

Schwachstellen

Alle genannten oder herausgefilterten Schwachstellen lassen sich sieben Kategorien zuordnen: Wissenssicherung, redaktionelle Prozesse, Inhaltsnutzung, Tools, Dokumentation, Kommunikation und Datenablage und -verwaltung. Die folgende Grafik veranschaulicht, wie oft die einzelnen Schwachstellenkategorien von den Befragten genannt bzw. Aussagen gemacht wurden. Eine Person konnte durch verschiedene Schwachstellen, die ihr einfielen, dieselbe Kategorie mehrmals nennen:

Anzahl der Aussagen pro Schwachstellenkategorie

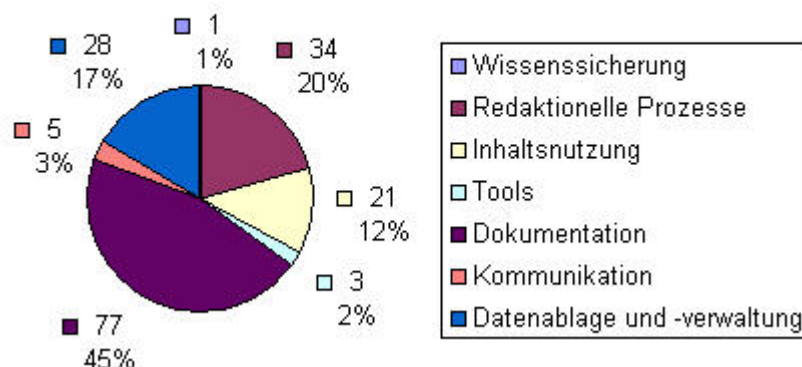


Abbildung 4-4: Anzahl der Aussagen pro Schwachstellenkategorie

Quelle: Eigene Erstellung

Mit 77 Nennungen halten die Befragten die Dokumentationen am meisten für verbesserungswürdig, gefolgt von den redaktionellen Prozessen innerhalb der Grid-Softwaregruppe.

Kriterien für optimale Dokumentationssituation

Anhand der Verbesserungswünsche, Anforderungen sowie Schwachstellen aus der Ist-Analyse lassen sich folgende Kriterien herausfiltern und gruppieren, die eine optimale Dokumentationssituation für die Grid-Softwaregruppe erfüllen soll:

Projektdokumentation

Angaben zur Projektdokumentation bezüglich Aktualität, Struktur, Informationsgehalt, Auffindbarkeit, Layout: Eine Projektdokumentation, deren Dokumente ...

- aktuell sind
- zeitnah publiziert werden

- über Änderungen bestehender Software informieren
 - für die verschiedenen Zielgruppen verständlich sind
 - kurz und strukturiert sind
 - Informationen schnell finden lassen
 - Inhalt und Zweck des Projektes kurz beschreiben
 - alle relevanten Informationen zum Projekt beinhalten
 - Angaben zu den von anderen verwendbaren Softwarekomponenten aufweisen
 - verlustfrei skalierbare Bilder haben
 - Anleitungen zur Software enthalten
 - über ausreichend Quellcodekommentierung verfügen
 - von zu Hause und weltweit erreichbar sind
 - im WWW schnell gefunden werden
 - mit der Corporate Identity des FZK übereinstimmen
 - den Layout- und Aufbaurichtlinien (z.B. von Verlagen, Konferenzen) entsprechen
 - die in einem Format verfügbar sind, das unabhängig von Tool und Betriebssystem verwendet werden kann
- ⇒ Zusammenfassend: Userfreundliche, für die Zielgruppen weltweit erreichbare Dokumentation

Erstellungsaufwand Angaben zur Vereinfachung der Erstellung: Unterstützung bei der Erstellung von Projektdokumentation durch ...

- Vorgaben zum Detaillierungsgrad, gerade in Bezug auf Zielgruppen
- Beispieldokumentation, an der sich der Autor orientieren kann („Templates, aus denen vorhandener Text einfach ersetzt wird“)
- Strukturvorgaben, in welcher Reihenfolge die Informationen aufzuführen sind
- Vorgaben, was alles erstellt und beachtet werden muss
- Angaben, wo benötigte Informationen gefunden werden
- eine Liste mit häufig verwendeten englischen Fachtermini
- Wiederverwendung von bestehenden englischen Inhalten
- entsprechende Tools

- ⇒ Zusammenfassend: Einfaches und schnelles Erstellen von Projektdokumentation ohne lange Einarbeitungszeit

Interne Datenhaltung

Angaben zur internen Datenhaltung: Datenhaltung, die ...

- für alle Beteiligten der Grid-Softwaregruppe zentral ist
- strukturiert und standardisiert ist
- einen schnellen Zugriff auf benötigte Informationen ermöglicht
- ⇒ Zusammenfassend: Zentrale Datenhaltung mit schnellem Zugriff auf Informationen

Besonderheiten

Angaben zu besonderen Anforderungen:

- Keine Änderung der Rechte bzw. Zuständigkeiten
- Wenn möglich, vorhandene oder kostenlose Tools benutzen
- Produkthaftung spielt keine Rolle, da die Grid-Softwaregruppe keine Haftung übernimmt. Produktion und Verkauf von Software gehört auch in nächster Zukunft nicht zu ihren Aufgaben.
- ⇒ Zusammenfassend: Keine Änderung der Rolle, kostenlose bzw. kostengünstige Tools, keine Produkthaftung

An die Dokumentation geknüpfte Ziele

Angaben zu Zielen, die an die Dokumentation geknüpft sind:

- das Sichern von Wissen trotz hoher Studentenfluktuation
- eine Verbesserung der internen Kommunikation in der Grid-Softwaregruppe und Partnerinstituten
- nicht als „Zweiter“ bei der Veröffentlichung neuer Entwicklungen den Urheberstatus verlieren
- am Forschungszentrum sowie auf dem nationalen und internationalen Markt einen hohen Bekanntheitsgrad haben
- Beteiligung an künftigen Forschungsprojekten sichern
- Präsentation der Entwicklungen nach Innen und Außen. Am FZK vor allem gegenüber den Instituten *IPE*, *IWR* und *Marketing*
- Überzeugungsarbeit leisten bezüglich des Nutzens der Serviceorientiertheit
- Leistungsfähigkeit als Wissenschaftler über seine Publikationen aufzeigen
- ⇒ Zusammenfassend: Optimale Präsentation nach Innen und Außen und Wissenssicherung

4.4 Ergebnis: Thema der Master-Thesis

Begründung für Softwarebewertungs-dokumentation

Bezogen auf die empfohlenen Dokumentationstypen (siehe Kapitel 4.2) zielt das Thema der Master-Thesis auf die Dokumentation zur *Softwarebewertung* ab. Die anderen wurden aufgrund folgender Ergebnisse aus der Ist-Analyse nicht ausgewählt:

- Die Dokumentation zum *Projektmanagement* ist vorhanden und wird unter Beachtung von Richtlinien und einem vorgegebenen Zeitrahmen erstellt. Sie bildet keine Dokumentationsschwachstelle.
- Dokumentation zur *Programmentwicklung* ist in Form von Studien-, Diplom- und Doktorarbeiten vertreten und stellt ebenfalls keine Dokumentationsschwachstelle dar.
- *Systemdokumentation*: Für die Dokumentation zur *Fehlerdiagnose* soll demnächst ein Bugtrackingsystem eingesetzt werden, während die Dokumentation zur Weiterentwicklung bereits durch Inline-Source-Kommentierung und den daraus generierten Webseiten mithilfe des Tools *JavaDoc* vorhanden ist. Sie stellt keinen Brennpunkt dar.
- *Benutzerdokumentation*: Die zur *Post-sales* Dokumentation gehörende Dokumentation zur Administration, Bedienung und Anwendungsprogrammierung ist kaum vorhanden und bildet eine Schwachstelle. Sie besitzt aber für die Grid-Softwaregruppe nicht dieselbe Relevanz wie die Softwarebewertungsdokumentation, da es keinen Verkauf und keine Produktion ihrer Software gibt (siehe Produkthaftung, S. 23).

Die im Bereich der *Pre-sales* Dokumentation angesiedelte Softwarebewertungsdokumentation erfüllt die mit der Dokumentation verbundenen Ziele wie die Präsentation der Entwicklungen nach Innen und Außen (siehe Ziele, S. 23) optimal, wobei im Fall der Grid-Softwaregruppe *Pre-offer* Dokumentation aufgrund des fehlenden Verkaufs treffend ist.

Ausschluss:
Printdokumentation

Die zur Softwarebewertungsdokumentation gehörenden wissenschaftlichen Publikationen, Poster und Flyer bleiben zwar durch die Online-Lösung unberücksichtigt, stellen aber nur bedingt eine Schwachstelle in der Dokumentation dar: Die als „ungeschriebene Gesetze“ vorhandenen Prozessabläufe werden über die interne Kommunikation gesteuert, weisen aber einige Schwachstellen auf. Es existieren jedoch Richtlinien zum Layout (Templates) und Aufbau, der zeitliche Rahmen ist von Außen durch Abgabe- bzw. Einreichungstermine vorgegeben und für ihre Erstellung stehen geeignete Tools bereit.

Brennpunkt:
Webauftritt

Ein Dokumentationsgebiet, was hingegen eher „brach“ liegt, stellt der zum Teil nur rudimentär mit Inhalten gefüllte Webauftritt der Grid-Softwaregruppe dar. Das fehlende Bewusstsein für den Internetauftritt bestätigte die Tatsache, dass nur

zwei der Befragten den Webauftritt von sich aus erwähnten, bei den anderen hingegen gezielt danach gefragt werden musste.

Das gewählte Thema, den Webauftritt der Grid-Softwaregruppe zu verbessern und die Softwareforschungsprojekte darin zu präsentieren, erfüllt die genannten Ziele wie die Präsentation der Entwicklungen nach Innen und Außen und zahlreiche Kriterien einer optimalen Dokumentationssituation (siehe S. 21):

Thema als Antwort auf die Anforderungen

Anforderungen	Lösung
Userfreundliche, für die Zielgruppen weltweit erreichbare Dokumentation	Konzeption eines Webauftrittes für die Grid-Softwaregruppe und für ihre Projekte
Einfaches und schnelles Erstellen von Projektdokumentation ohne lange Einarbeitungszeit	■ Automatisierungsprozesse ■ Erstellung eines Redaktionsleitfadens
Zentrale Datenhaltung mit schnellem Zugriff auf Informationen	Ordnerstrukturkonzept
Verwendung vorhandener oder kostenloser Tools	Aufbau einer Technologie-Architektur aus bekannten und neuen kostenlosen/kostengünstigen Tools
→ Optimale Präsentation nach Innen und Außen und Wissenssicherung	

Tabelle 4-1: Thema als Antwort auf die Anforderungen

Quelle: Eigene Darstellung

CMS des FZK

Für seinen Internetauftritt benutzt das FZK seit 2003 das Content Management-System von *Stellent*. Die Webseiten werden mit einem Word-Editor unter Einsatz von Templates und Formatvorlagen geschrieben und in HTML-Webseiten konvertiert.



Abbildung 4-5: Erstellung einer Webseite mit Word-Editor

Quelle: Dokument und Formatvorlagen von Bohrmann (2007)

Bisher hat die Grid-Softwaregruppe ihren Webauftritt nicht über das CMS des FZK zur Verfügung gestellt.

Gründe gegen CMS Verwendung

Die Entscheidung, weiterhin nicht das CMS zu nutzen, basiert auf den beiden folgenden Gegebenheiten:

- Die Grid-Softwaregruppe besitzt keinen direkten Zugriff auf das CMS und damit auf den Editor zur Texterstellung.
- Die Grid-Softwaregruppe kann zwar Texte zur Verfügung stellen, aber nicht selbst veröffentlichen oder aktualisieren. Vom Weiterleiten an den Webmaster bis zum Veröffentlichen der neuen Inhalte vergeht eine zu große Zeitspanne. Gerade in der Schnellebigkeit von Entwicklungen und hinsichtlich der Notwendigkeit der Forscher, ihre Projekte zeitnah zu dokumentieren, ist dies eine inakzeptable Lösung.

Projekte im Internet am FZK

Ein Blick zu anderen Forschungsgruppen am IPE und sieben weiteren Instituten am FZK¹⁰ zeigt hinsichtlich der Projektdokumentation folgendes Bild:

- Eine einheitliche Projektbeschreibung im WWW existiert weder instituts- noch abteilungs- oder projektübergreifend – unabhängig davon, ob das Projekt mithilfe des CMS von *Stellent* erstellt wurde oder nicht.
- Ob das CMS für eine Projektdokumentation eingesetzt wird, ist von Institut zu Institut unterschiedlich (z.B. IAI, IBG), aber auch zwischen den Abteilungen bzw. Fachgruppen eines Institutes (z.B. IRS).
- Manchmal sind Projekte durch keine Webseiten vertreten (z.B. IPE, IKET).
- Die Projektpräsentationen innerhalb einer Abteilung bzw. Fachgruppe unterscheiden sich stets in Informationsgehalt, Seitenaufbau und -länge. Sie reichen von einer Webseite mit einem Ansprechpartner und drei weiterführenden Links über eine kurze Projektbeschreibung bis hin zu unterschiedlichsten Informationen gepaart mit vielen Bildern. Strukturvorgaben für einen einheitlichen Webseitenaufbau einer Projektbeschreibung gibt es somit im CMS nicht. Lediglich die Corporate Identity wird im Gegensatz zu den erstellten Webseiten ohne CMS gesichert (z.B. ITP, IBG, IKET).
- In der Regel wird eine Webseite angeboten. Dabei wird häufig versucht, alle Informationen zum Projekt aufzuführen. Daraus ergeben sich lange Webseiten mit zahlreichen Verlinkungen in alle „Himmelsrichtungen“ im Fließtext (z.B. IKET, IBG).
- Ein Institut (IAI) präsentiert seine Projekte in einem selbst entworfenen Auftritt, der versucht die Corporate Identity des FZK aufzunehmen.

¹⁰IAI, IWR, ITP, IK, IKET, IBG, IRS

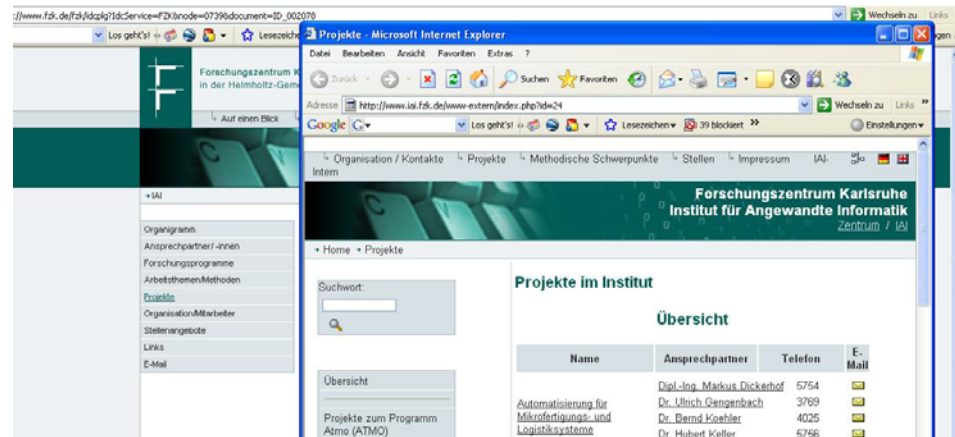


Abbildung 4-6: Webauftritt für die Projekte des IAI

Quelle: <http://www.iai.fzk.de/www-extern/> [Zugriff: 20.05.2007, 20:30 MESZ]

- Abteilungen bzw. Fachgruppen präsentieren ihre Projekte zum Teil auch ohne Anpassung an die Corporate Identity (z.B. IK, IRS).

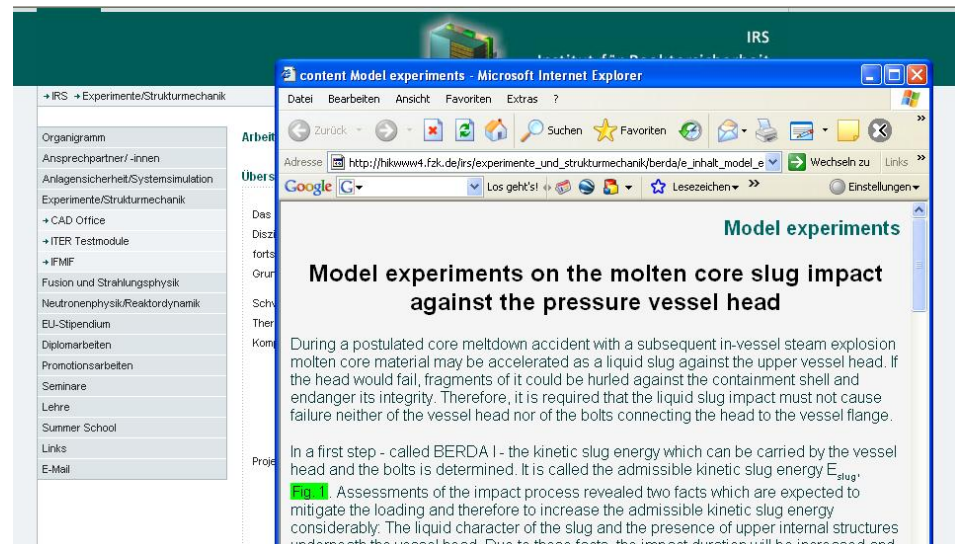


Abbildung 4-7: Eine Projektbeschreibung bei IRS

Quelle: <http://www.fzk.de/fzk/idcplg?IdcService=FZK&node=0397> [Zugriff: 10.09.2007 09:30]

⇒ Zusammenfassend:

Eine standardisierte, userfreundliche Projektbeschreibung für das WWW existiert nicht einmal projektübergreifend. Der Einsatz des CMS unterstützt lediglich ein einheitliches Erscheinungsbild. Die Gründe der Grid-Softwaregruppe ihre Inhalte nicht über das CMS einstellen zu lassen, sind bei anderen Fachgruppen ebenfalls vorhanden. Dies bestätigt, eine eigenständige Lösung für den Online-Auftritt eines Projektes zu entwickeln und dies ohne den Einsatz des CMS.

5 Technologie-Architektur

5.1 Einführung

In den folgenden beiden Kapiteln wird zum einen beschrieben, unter Einsatz welcher Methoden die einzelnen Technologien ausgewählt wurden. Zum anderen wird ein Überblick zu den festgelegten Technologien und ihren Einsatzgebieten für die Erstellung des Webauftrittes gegeben.

5.2 Vorgehensweise bei der Definition der Technologie-Architektur

Erkenntnisse der Ist-Analyse

Den Ausgangspunkt bildeten die gewonnenen Erkenntnisse aus der Ist-Analyse zur bestehenden Technologie-Architektur:

- Für die *Veröffentlichung* der Webseiten steht eine gut funktionierende Technologie-Architektur bereit.
- Zahlreiche Tools dienen der *Erstellung* von Printdokumentation und Grafiken sowie der Bildbearbeitung.
- Lediglich ein geeigneter Editor für die Webseitenerstellung und ein Versionierungssystem für die Dokumentation fehlen, was bemängelt wurde. Bisher haben die Softwareentwickler ihre Webseiten mit den beiden Tools *Editor* und *OOo Writer* selbst in HTML erstellt und jeder hat seine Dokumente selbst verwaltet und nur nach Bedarf versioniert.

Zunächst wurden die benötigten Technologiearten für den Erstellungsprozess des Webauftrittes identifiziert:

- Editor(en) zur Formelerstellung
- Editor(en) zur Grafikerstellung
- Editor(en) zur Bildbearbeitung
- Editor(en) zur Webseitenerstellung
- Versionierungssystem

Gleichzeitig wird ein Editor für die Erstellung von Webseitenvorlagen benötigt.

Vorhandene Tools nutzen

Die von der Gruppe eingesetzten Publikationstechnologien und die erprobten und benutzten Editoren für die Formel-, Grafik- und Bilderstellung sollen auch weiterhin verwendet werden. Die Vorteile:

- Der Autor kann mit seinen ihm vertrauten Tools arbeiten und muss keine neuen Kenntnisse erwerben.
- Der Installationsaufwand auf den PCs entfällt.

XML-Editor

Die Wahl, einen XML-Editor als Autorenwerkzeug für die Webseitenerstellung zu verwenden, erfolgte aufgrund der zahlreichen Vorteile von XML und XSL:

- XML produziert neutrale toolunabhängige Daten, so dass ein Autor nicht an einen speziellen Editor gebunden ist.
- Zudem ermöglicht die Kombination aus XML und XSL, Single Source und Cross Media Publishing umzusetzen und die damit verbundenen Vorteile zu nutzen. Dies unterstützt den Wunsch der Gruppe, ihren Schreibaufwand zu minimieren.
- Die Struktur eines Dokumentes wird dem Autor mithilfe von Element- und Attributnamen vorgegeben und unterstützt ihn in einem konsistenten Webseitenaufbau. Die Namen können semantisch gewählt werden und sind dadurch leicht verständlich.

Mit dem Technologie-Verantwortlichen der Grid-Softwaregruppe wurde zunächst eine Liste an XML-Editoren zusammengestellt.

Es wurden zwei Pflichtkriterien für eine erste Auswahlrunde der verschiedenen Tools festgelegt:

- Einsetzbar auf den Betriebssystemen Windows und Linux
- Kostenloser oder kostengünstiger (bis 200 Euro) Editor

Bewertungskriterien für Webseiteneditor

Es folgte die Aufstellung von weiteren Bewertungskriterien für den XML-Editor, die mit dem Technologie-Verantwortlichen der Grid-Softwaregruppe abgeglichen und anschließend gewichtet wurden. Wenn nötig wurde zur Gewichtung die Präferenzmatrix von Kirchhofer eingesetzt. Ziel war es, nicht nur die Ansichten eines Redakteurs mit einfließen zu lassen, sondern auch die eines IT-Beauftragten (siehe Anhang B).

In einer zweiten Auswahlrunde wurden aus den verbliebenen Editoren anhand von weiteren Pflichtkriterien wie eine *Formularansicht* oder die *sofortige Überprüfung der eingefügten Strukturelemente auf Korrektheit* drei Editoren selektiert: XMLmind Personal Edition, oXygen XML Editor und Altova Authentic.

Bei diesen wurde im Selbsttest überprüft, inwieweit sie die anderen Bewertungskriterien erfüllen. Mithilfe der Angebotsbewertung von Kirchhofer und den Ergebnissen aus dem Selbsttest wurde der am besten geeignete Editor ausgewählt.

Die Festlegung, auf jedem PC die Tools für die Webseitenerstellung zu installieren, wurde zugunsten einer virtuellen Maschine aufgegeben. Ihre Vorteile sind:

- Der mehrfache Installations- und Konfigurationsaufwand entfällt.
- Jeder Autor arbeitet mit derselben XML-Editor-Version.

- Updates müssen nur einmal durchgeführt werden.

Versionierungssystem Für die Auswahl eines Versionierungssystems wurde aus den beiden vorhandenen Systemen *Subversion* und *Concurrent Versions System* das passende ebenfalls anhand von Bewertungskriterien ausgesucht. Beide werden für die Codekommentierung bereits erfolgreich von der Gruppe eingesetzt, eignen sich aber auch für die Dokumentation (siehe CD Datei *Technologie_Bewertungen.xls*).

5.3 Ergebnis der Technologie-Architektur

Folgende Editoren stehen den Autoren künftig für den Webauftritt zur Verfügung:

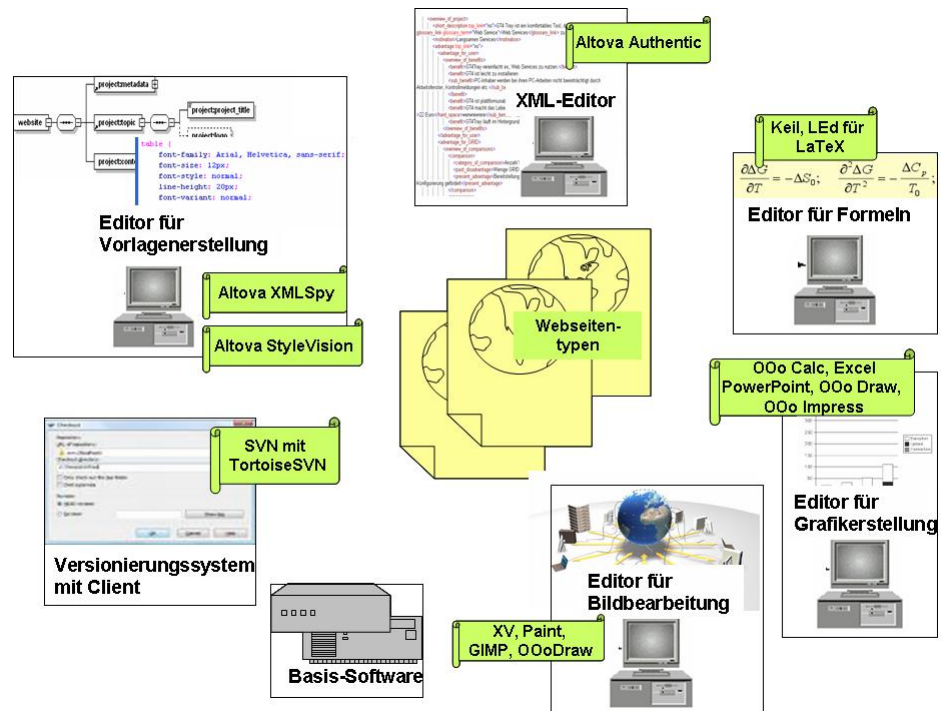


Abbildung 5-1: Editoren der Technologie-Architektur

Quelle: Eigene Darstellung

Insgesamt sind an den redaktionellen Prozessen vom Erstellen bis hin zum Publizieren und Archivieren folgende Technologien beteiligt, die die angegebenen Funktionen beim Webauftritt übernehmen:

Eingesetzte
Technologien im
Redaktionsprozess

Technologien	Einsatz/Funktion	Lokalisation
Virtuelle Maschine	... ist ein physisch nicht existierender PC, an dem sich der User anmeldet, um mithilfe der dortigen Tools die Webseite zu erstellen und um auf das Versionierungssystem zuzugreifen.	Windows-Server
Altova Authentic 2008 Desktop Edition	... ist der Editor, mit dem in WISIWYG- oder Formularansicht die XML-Quelldatei für eine Webseite mit Inhalt gefüllt wird.	Virtuelle Maschine
Saxon 8.9	... ist der Prozessor, mit dem die Transformation von XML in HTML mithilfe von Stylesheets in XSLT 2.0 vorgenommen wird.	Virtuelle Maschine

Technologien	Einsatz/Funktion	Lokalisation
Altova XMLSpy 2005 Enterprise Edition	... ist der Entwicklungseditor, mit dem XML-Schemas und XSLT-Stylesheets erstellt werden. Diese sind für die Strukturvorgabe einer Webseite und ihr Aussehen verantwortlich.	Lokal auf Windows-PC
Altova StyleVision 2005 Enterprise Edition	... ist der Entwicklungseditor, mit dem Stylesheets speziell für den Authentic-Editor erstellt werden. Ein solches Stylesheet ist notwendig, um die XML-Quelldatei einer Webseite dort anzeigen und mit Inhalt befüllen zu lassen.	Lokal auf Windows-PC
PuTTY	... ist der Keygenerator, mit dem der User einen Schlüssel nach dem Zufallsprinzip erzeugt und ihm ein Passwort zuweist. Beides wird für den Service Pageant benötigt.	Virtuelle Maschine
Pageant	... ist der Service, der beim Ein- und Auschecken von Inhalten in SVN über TortoiseSVN die jeweils 3malige Identifikation via Passwortabfrage übernimmt. Der User gibt dazu bei Pageant nach dem Einloggen in die virtuelle Maschine sein Passwort an. Pageant übernimmt die Identifikation, solange der User auf der virtuellen Maschine eingeloggt ist.	Virtuelle Maschine
TortoiseSVN 1.4.3	... ist der Client, über den der User auf Subversion zugreift, um seine Dateien, Ordner oder die gesamte Verzeichnisstruktur (Repository) ein- und auszuchecken. Er kann sich zudem in begrenztem Rahmen die Metadaten der einzelnen Inhalte anzeigen lassen.	Virtuelle Maschine
Subversion (SVN) 1.4.5	... ist das Versionierungssystem, mit dem die Softwaregruppe sowohl ihren Webauftritt als auch die dazugehörigen Quelldateien bzw. Bilder und Grafiken versioniert.	Virtuelle Maschine
Subversion-Server	... ist der Apache-Server, auf dem Subversion installiert ist.	-
viewCVS	... ist eine Benutzeroberfläche, mit deren Hilfe sich der User die in Subversion verwaltete Verzeichnisstruktur im Browser anzeigen lassen kann. Er erhält zu den einzelnen Dateien und Ordnern die über TortoiseSVN angegebenen Default-Metadaten als kurze Beschreibung. Dazu zählen z.B. die Versionsnummer oder das Datum des Eincheckvorganges. Auch lassen sich die Unterschiede zwischen den Versionen anzeigen.	Virtuelle Maschine
MikTeX 2.2	... ist das LaTeX Softwarepaket für Windows.	Virtuelle Maschine
LEd/Keil	... sind zwei Editoren für LaTeX, mit denen der User eine mathematische Formel erstellt.	Lokal auf dem PC des Autors
JabRef 2.3 beta2	... ist ein Editor, mit dem der User die Textdatei mit allen Publikationsangaben (BibTeX-Datei) erstellt.	Lokal auf dem PC des Autors
BibTeX	... ist ein Programm, mit dem die Gruppe aus einer Textdatei, in der all ihre Publikationen in einer bestimmten Syntax aufgelistet sind (BibTeX-Datei), eine Webseite zu ihren Publikationen erzeugt. Das Ausgabeformat und Layout regelt eine spezielle BibTeX-Stildatei.	Lokal auf dem PC des Autors
OOo Calc/ Excel	... sind die Editoren, die der User zur Tabellenkalkulation einsetzen kann, abhängig vom Betriebssystem seines PCs. Sie dienen zum Erzeugen eines Diagramms (Linien-/Balkendiagramm).	Lokal auf dem PC des Autors
PowerPoint/ OOo Impress/ OOo Draw	... sind die Editoren, die der User für die Erstellung von Grafiken einsetzt.	Lokal auf dem PC des Autors

Technologien	Einsatz/Funktion	Lokalisation
XV/Paint/GIMP 2.2.17 /OOo Draw	... sind die Editoren, die der User zur Bildbearbeitung einsetzt, z.B. um ein Bild für die Webseite zurechtzuschneiden.	Lokal auf dem PC des Autors
SSH Secure Shell (TM) 3.2.5	... ist ein Dateitransferprogramm (FTP), mit dem der User die Verbindung zum Webserver via Passwort-eingabe herstellt und die freigegebenen Webseiten und Bilder auf den Webserver überträgt.	Virtuelle Maschine
super-sed 3.6.2	... ist ein Editor, der im Batch- oder Shell-Betrieb läuft, d.h. er ist kein interaktiver Editor. Er wird für die Automatisierungsprozesse benötigt. Die Batch-Dateien können mit seiner Hilfe Dateien bezüglich ihres Textes ändern.	Virtuelle Maschine
Image Magick 6.3.5.6	... ist ein Bildbearbeitungsprogramm, das für einen Automatisierungsprozess benötigt wird, bei dem eine Formel im TeX-Format in ein Bild im PNG-Format umgewandelt wird.	Virtuelle Maschine
Webserver (fuzzy.fzk.de)	... ist der Server, auf dem sich der im WWW veröffentlichte Webauftritt der Grid-Softwaregruppe befindet.	-

Tabelle 5-1: Technologien und ihre Funktion im Redaktionsprozess

Quelle: Eigene Darstellung

Mit der Auswahl des XML-Editors *Authentic* können zugleich die beiden bereits vorhandenen, aber bisher nicht benutzten Tools *Altova XMLSpy* und *Altova StyleVision* als Entwicklungsumgebung für den Informationsdesigner eingesetzt werden. Der Hersteller *Altova* sieht den ausgesuchten Webseiteneditor *Authentic* zwar unter Linux als einsetzbar an, es zeigte sich aber, dass dies nur durch Einbettung des Editors in eine *Wine* -Umgebung möglich ist, die eine Windows-Oberfläche simuliert. Dies erfordert jedoch eine aufwändige Konfiguration und lief zudem auf dem vorhandenen System nicht stabil. Der Einsatz der virtuellen Windows-Maschine löst dieses Problem.

Hinsichtlich der Editoren im Grafik- und Bildbereich wurden Empfehlungen gegeben, um neuen Mitarbeitern und Studenten eine Hilfestellung zu bieten.

6 Automatisierungsprozesse

6.1 Einführung

Die folgenden Kapitel beschäftigen sich mit denjenigen redaktionellen Prozessen, deren Automatisierung sinnvoll erschien und die realisiert wurden (siehe Anhang D).

6.2 Vorgehensweise bei Konzeption und Realisierungsphase

Automatisierungskriterien	Bei der Definition der Prozesse wurden folgende Punkte überprüft: ▪ Muss ein Prozess standardisiert ablaufen? ▪ Ist ein Prozess besonders fehleranfällig? ▪ Kann dem User die Arbeit durch Automatisierung erleichtert werden?
Zusammenspiel: Authentic, Batch, XSLT	Traf einer der Punkte zu, wurde versucht diesen Prozess mithilfe der Konfigurationsmöglichkeiten des Editors <i>Authentic</i> und mithilfe von Batch-Dateien zu automatisieren. Falls nötig wurde außerdem eine Transformation mit XSLT hinzugefügt.
Konzeptbeschreibung	Das Konzept für einen Prozess wurde in Tabellenform detailliert dokumentiert (siehe CD Datei <i>Automatisierungsmöglichkeiten.xls</i>). Darin sind beschrieben: ▪ die Bewertungsgrundlage für den Automatisierungsprozess ▪ die Prozessbeschreibung aus Sicht des Prozessverantwortlichen ▪ die Prozessbeschreibung aus Sicht des Informationsdesigners
Relevanz	Um die Relevanz eines Automatisierungsprozesses zu begründen, stellte die Autorin in der Bewertungsgrundlage den Nutzen und die Einwände einander gegenüber. Anhand der Gesamteinschätzung wurde festgelegt, in welcher Reihenfolge die Prozesse umgesetzt werden. Aufgrund der großen Anzahl konnten innerhalb des Zeitrahmens nicht alle Prozesse realisiert werden.
Prozessbeschreibung	Zudem erhält jeder Prozess eine Beschreibung mit grundlegenden Informationen zum Ablauf wie z. B. Beginn und Ende, erforderliche Kenntnisse bezüglich der eingesetzten Tools oder die einzelnen Handlungsschritte.
Detaillierte Angaben zur technischen Umsetzung	Für einen schnellen Überblick zum Zusammenspiel der verschiedenen Dateien bei einem Prozess ist die technische Umsetzung übersichtlich beschrieben: Der Informationsdesigner findet ... ▪ eine Auflistung der Programmdateien mit ihren Hauptfunktionen und ihrem Zusammenspiel

- die technischen Voraussetzungen bzw. den Input, damit der Prozess funktioniert
- die Teilfunktionen und -ergebnisse jeder Programmdatei
- den Interaktionsprozess Anwender-Maschine mit Vorgaben zur Batch-Programmierung oder - soweit wie möglich - zur JavaScript-Programmierung für die *Authentic*-Konfigurationsdateien

Konsistente Codekommentierung

Zusätzlich zur Beschreibung der Prozesse wurden die einzelnen Batch-Dateien in festgelegter Form kommentiert. Dadurch sind Ziel, Zweck, Input und Output eines Codeabschnittes sowie wichtige Schritte schnell ersichtlich.

```
Rem *****
Rem Ziel: Anführungszeichen in den werten der globalen variablen entfernen
Rem Zweck: Korrekter Programmverlauf
Rem Input: Werte der globalen Variablen aus vorherigem Programmschritt
Rem Output: Globale Variablen mit angepassten werten für die folgenden Programmschritte
Rem Schritte/Erläuterung:
Rem - Nimm alten Variablenwert und gebe ihn ab dem 2.Zeichen aus, entferne aber das letzte
Rem   Zeichen des Wertes
Rem *****
```

Abbildung 6-1: Beispiel für Codekommentierung in standardisierter Form

Quelle: Batch-Datei *check_userinput_call_generate_project.bat*

Bewertung von Alternativen

Für den Prozess, eine mathematische Formel im *TeX*-Format in ein Bild im *PNG*-Format umzuwandeln, wurden die verschiedenen möglichen Ansätze mithilfe von Bewertungskriterien und der Angebots- und Präferenzmatrix von Kirchhofer gegeneinander abgewogen, um eine optimale Automatisierung zu erzielen (siehe Anhang C).

6.3 Automatisierungskonzept und technische Realisierung

Folgende Automatisierungsprozesse wurden definiert:

Automatisierungsprozesse	Realisierung durch ...
Neues Projekt anlegen	Authentic/Batch
Userdaten abfragen	Authentic
Glossartermdatei des Projektnamens füllen	Authentic/Batch
Glossarterm anlegen	Authentic/Batch
Links auf Glossartermdateien setzen	Authentic/Batch/XSLT
Glossar generieren	Authentic/Batch/XSLT
Webseitentypen <i>Aktuelle Projekte</i> oder <i>Alte Projekte</i> automatisch generieren	Authentic/Batch/XSLT
Formel im <i>TeX</i> -Format in Bild im <i>PNG</i> -Format umwandeln	Authentic/Batch
Alle XML-Dateien in PHP/HTML umwandeln	Authentic
Datei als <i>created</i> markieren	Authentic
Datei als <i>modified</i> markieren	Authentic
Datei als <i>reviewed</i> markieren	Authentic
Datei als <i>approved</i> markieren	Authentic
Datei als <i>published</i> markieren	Authentic
Datei als <i>archived</i> markieren	Authentic

Automatisierungsprozesse	Realisierung durch ...
Metadatendatei für neue Bilder anlegen	Authentic/Batch
Webseite auswählen	Authentic
Metadatendatei für gesamten Webauftritt generieren	Authentic/Batch/XSLT
Reminder, für ein neues Bild eine Metadatendatei erzeugen	Authentic/Batch

Tabelle 6-1: Übersicht zu den Automatisierungsprozessen

Quelle: Eigene Darstellung

SED	Eine wichtige Rolle bei der Automatisierung spielte der Einsatz des <i>StreamEditors (SED)</i> . Seine Kommandos und Patterns bieten zahlreiche Möglichkeiten Texte zu bearbeiten und wurden direkt in die Batch-Dateien eingebaut. Die Batch-Programmierung weist für Textänderungen nur ein eingeschränktes Repertoire auf.
Globale Variablen	Bei der Programmierung der Batch-Dateien wurde darauf geachtet, Dateipfade oder Angaben, die mehrmals im Dokument verwendet werden, am Dokumentanfang in globalen Variablen zu speichern.
Auszug aus den Prozessen	Im Folgenden werden nur diejenigen Prozesse ausführlicher erläutert, die von der Autorin umgesetzt wurden. Die restlichen Prozesse sind in Form von Prozessbeschreibungen dargestellt. Für diese stellte die Autorin die entsprechenden Templatedateien bereit. Die Programmierung des <i>Authentics</i> wurde von der Autorin angefangen, aufgrund des Zeitaufwandes jedoch von einem Mitarbeiter der Grid-Softwaregruppe nach ihren Konzeptvorgaben fertig gestellt. Die Umsetzung von Prozessen mit XSLT ist in den Kapiteln 8 und 9 beschrieben.
6.3.1 Automatisierungsprozess: Neues Projekt anlegen	
Ziel der Automatisierung	Bei Projektbeginn lässt sich der User aus einem Prototyp-Projektordner seine eigenen, speziell auf sein Projekt zugeschnittenen Ordner generieren.
Zweck der Automatisierung	Die Automatisierung sichert einen konsistenten Aufbau jedes einzelnen Projektordners sowie eine standardisierte Datei- und Ordnerbenennung. Der Softwareentwickler erhält automatisch alle Webseitendateien, die für sein Projekt ange-dacht sind. Dadurch werden keine Webseiten vergessen und der Webauftritt weist dieselbe Struktur und den gleichen Informationsgehalt auf.
Prozessablauf	Sobald der User die Prototyp-Projektdatei im <i>Authentic</i> öffnet, erscheint ein Eingabeformular und fordert ihn auf, Angaben zu seinem Projekt einzutragen. Mithilfe von Kontrollmechanismen und Auswahlboxen wird eine korrekte Eingabe gesichert und der User durch Tooltips geleitet. Im Anschluss wird die Projektmappe generiert, an entsprechenden Stellen abgelegt und die Projektdatei für sein Pro-

Technische
Realisierung

jekt öffnet sich im *Authentic*. Meldungen informieren den User über den Stand der Generierung und in welchen elektronischen Ordnern sein Projekt zu finden ist.

Die technische Durchführung erfolgt im Zusammenspiel des Prototyp-Projektordners mit drei Batch-Dateien und Skripten im *Authentic*. Ursprünglich wurde eine Batch-Datei so programmiert, dass das Anlegen eines neuen Projektes ohne Eingabe über den *Authentic* und stattdessen über Eingabe im DOS-Fenster funktioniert und die Kontrollmechanismen realisiert.

Mithilfe der anderen beiden Batch-Dateien werden ein Projektordner mit allen benötigten Unterordnern und Quelldateien sowie ein weiterer für den Output inklusive Unterordner entsprechend des Projekttyps automatisch angelegt. Datei- und Ordnernamen werden an den Projektnamen angepasst, in den XML-Dateien Metadaten-Elemente und andere mit den Userangaben befüllt und Dateipfade geändert, wie z. B. der Schemapfad. Besonders bei der *Authentic*-Projektdatei werden zahlreiche Anpassungen durchgeführt.

Der Prototyp-Projektordner enthält alle für ein Projekt notwendigen Ordner und Dateien wie XML-Templates für die Webseiten oder die *Authentic*-Projektdatei. Um über die Batch-Dateien diejenigen Ordner und Dateien herauszufiltern, die zum Projekttyp gehören, wurden die Ordner und Dateien nach bestimmten Namenskonventionen benannt und spezielle Textdateien in den Ordnern angelegt:

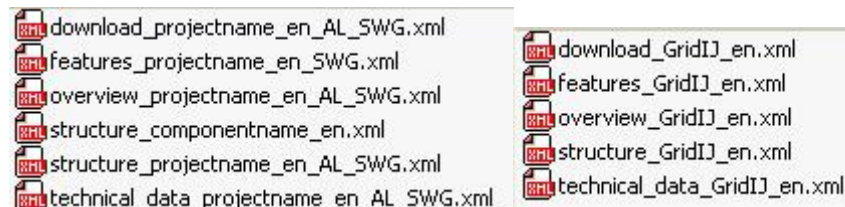


Abbildung 6-2: Templatenamen vs. angepasste Dateinamen für Projekt *GridIJ*

Quelle: Eigene Darstellung

- Die Abkürzungen AL und SWG stehen für den jeweiligen Projekttyp im Dateinamen. Ist er vorhanden, wird die übernommene Datei nicht gelöscht.
- Dateien, die erst während des Erstellungsprozesses der Webseiten anfallen, werden vom Autor bei Bedarf per Buttonklick im *Authentic* eingefügt (z. B. die Datei *structure_component_en.xml*). Deshalb enthält der Dateiname keine Projektabkürzungen, so dass die kopierte Datei ebenfalls gelöscht wird.
- Für den Fall, dass ein Ordner nicht zum Projekttyp gehört, wird eine leere Textdatei angelegt. Der Name *delete_for_AL* bedeutet, dass der Unter-

ordner nicht im Projektordner bleibt. Diese Technik wird z.B. beim Ordner *online_help* eingesetzt:

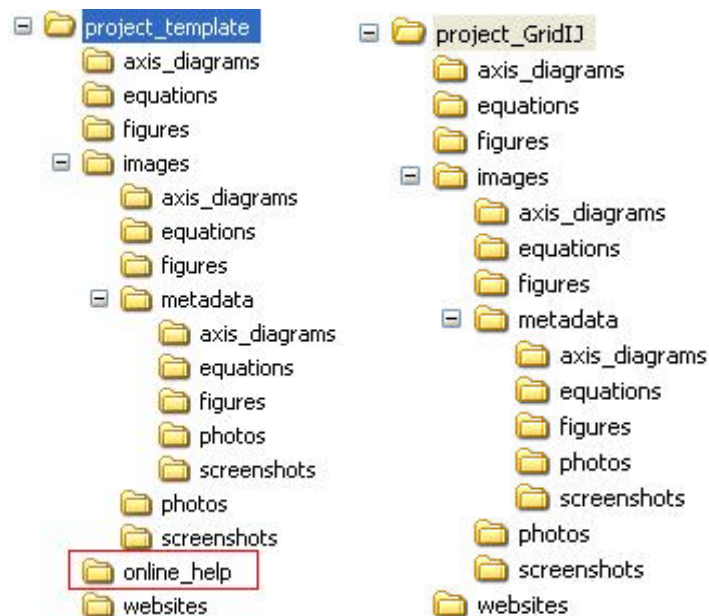


Abbildung 6-3: Prototyp-Projektordner vs. Projektordner für Projekt *GridIJ*

Quelle: Eigene Darstellung

Um die Userangaben an den entsprechenden Stellen in den Ordner- und Dateinamen oder in den XML-Templates über die Batch-Dateien einzusetzen, wurden spezielle Platzhalter wie *projectname* oder *\$PROJECT_END\$* gesetzt:

```
<project:metadata>
<project:project_data>
  <project:project_type>$PROJECT_TYPE$</project:project_type>
  <project:project_period>
    <project:start>$PROJECT_START$</project:start>
    <project:end>$PROJECT_END$</project:end>
  </project:project_period>
  <project:project_execution>
    <project:project_collaborator>
      <project:first_name>$CREATOR_FIRSTNAME$</project:first_name>
      <project:last_name>$CREATOR_LASTNAME$</project:last_name>
    </project:project_collaborator>
  </project:project_execution>
  <project:project_supervisor>
    <project:first_name>$SUPERVISOR_FIRSTNAME$</project:first_name>
    <project:last_name>$SUPERVISOR_LASTNAME$</project:last_name>
  </project:project_supervisor>
  <project:project_title>$PROJECT_TITLE$</project:project_title>
</project:project_data>
```

Abbildung 6-4: XML-Template mit Platzhaltern

Quelle: Codeausschnitt aus Templatedatei *overview_project.xml*

```

<project:metadata>
<project:project_data>
  <project:project_type>student trainee project</project:project_type>
  <project:project_period>
    <project:start>2006-10</project:start>
  </project:project_period>
  <project:project_execution>
    <project:project_collaborator>
      <project:first_name>Alexander</project:first_name>
      <project:last_name>Frank</project:last_name>
    </project:project_collaborator>
  </project:project_execution>
  <project:project_supervisor>
    <project:first_name>Rainer</project:first_name>
    <project:last_name>Stotzka</project:last_name>
  </project:project_supervisor>
  <project:project_title>GridIJ</project:project_title>
</project:project_data>

```

Abbildung 6-5: XML-Template für Projekt *GridIJ*Quelle: Codeausschnitt aus XML-Projektdatei *overview_GridIJ_en.xml*

Der Aufruf einer Batch-Datei mit Parameterübergabe wird im *Authentic* mithilfe von JavaScript durch folgenden Codeabschnitt geregelt:

```

wshShell = new ActiveXObject ("WScript.Shell");
var oExe=wshShell.Run ("Batchdateiname.bat " + Parametername);

```

Abbildung 6-6: Aufrufbeispiel für eine Batch-Datei mit Parameterübergabe

Quelle: Eigene Darstellung

6.3.2 Automatisierungsprozess: Links auf Glossartermdateien setzen

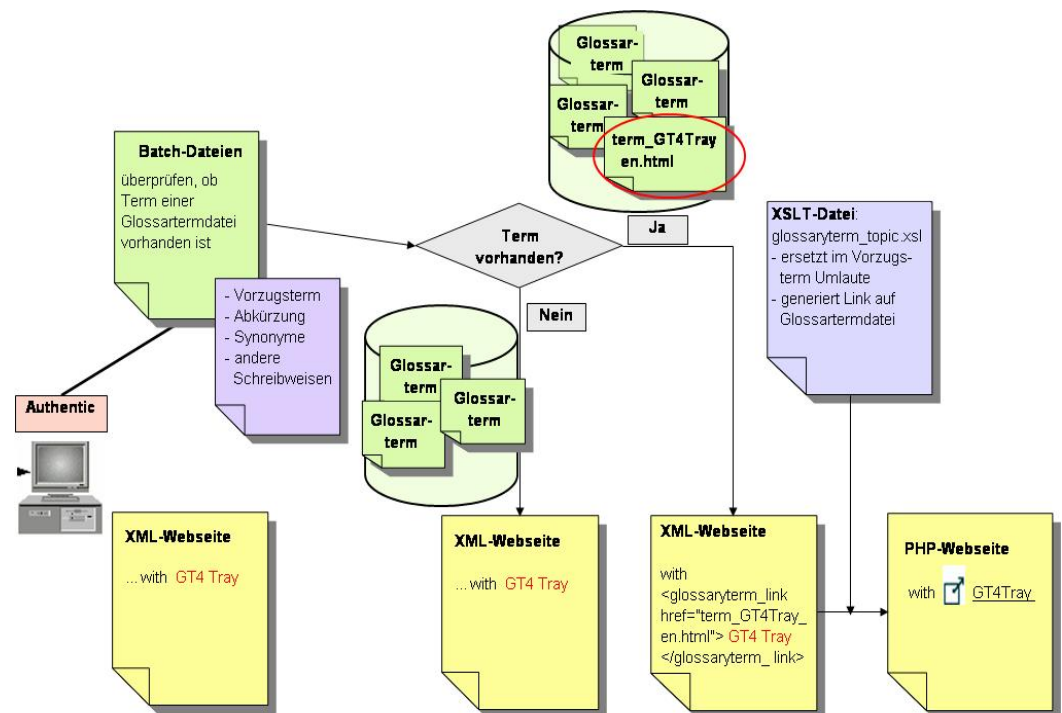
Ziel der
Automatisierung

Bei der fertig gestellten XML-Datei für eine Webseite werden automatisch Links auf die Dateien der veröffentlichten Glossarterme gesetzt, sobald ein Wort gefunden wird, das im Glossar als Vorzugsbenennung, Abkürzung oder Synonym aufgeführt ist. Auch verschiedene Schreibweisen werden berücksichtigt.

Zweck der
Automatisierung

Durch die Automatisierung entfällt für den User das zeitaufwändige Überprüfen, ob Benennungen in seiner Webseite als Glossarterme geführt werden. Gleichzeitig werden die Glossarterme optimal genutzt und ein User ohne Vorkenntnisse beim Lesen der Webseiten unterstützt. Auch werden „tote“ Links vermieden, da die Linkpfade nicht manuell gesetzt werden.

Prozessablauf

Abbildung 6-7: Prozessablauf *Links auf Glossartermdateien setzen*

Quelle: Eigene Darstellung

Der Autor und Freigeber starten den Prozess per Buttonklick und können über Eingabeformulare und Dateidialoge auswählen, ob in allen oder nur in bestimmten Dateien Links gesetzt werden sollen. Tooltips und Kontrollmeldungen zum Prozessstand leiten ihn. Zum Einsatz der XSLT-Datei siehe Kapitel 9.6.1.

Technische
Realisierung

Mithilfe von zwei Batch-Dateien werden in den übergebenen XML-Dateien alle bereits vorhandenen Verlinkungen auf Glossarterme entfernt. Alle Glossartermdateien werden herausgefiltert, die bereits veröffentlicht sind. Für jede Glossartermdatei werden die angegebenen Benennungen selektiert und in eine separate Textdatei geschrieben. Danach wird für jede XML-Datei entsprechend ihrer angegebenen Sprache (de/en) überprüft, ob sie die Benennungen enthält und bei Übereinstimmung der Link mit relativem Pfad – entsprechend der Ausgangsposition der XML-Datei – sowie der Linktyp und Linktext eingetragen. Alle Angaben werden automatisch generiert.

Voraussetzung für diese Mechanismen sind konsistente Glossartermdateinamen, standardisierte Ordnerstrukturen bzw. -namen sowie der Default-Wert „-“ bei nicht angegebenen Benennungen in einer Glossartermdatei. Insgesamt müssen pro Glossartermdatei acht Angaben zu den Benennungen vorhanden sein: vier zum deutschsprachigen und vier zum englischsprachigen Eintrag:

```

<project:entry project:language="de">
  <project:term project:preferred="yes">GT4Tray</project:term>
  <project:abbreviation project:preferred="no"></project:abbreviation>
  <project:synonyms></project:synonyms>
  <project:different_notations>GT4 Tray</project:different_notations>
  <project:definition><project:textsection>GT4 Tray ist ...</project:textsection></project:definition>
  <project:sources>
    <project:source/>
  </project:sources>
</project:entry>
<project:entry project:language="en">
  <project:term project:preferred="yes">GT4Tray</project:term>
  <project:abbreviation project:preferred="no"></project:abbreviation>
  <project:synonyms></project:synonyms>
  <project:different_notations>GT4 Tray</project:different_notations>
  <project:definition><project:textsection>GT4 Tray is...</project:textsection></project:definition>

```

Abbildung 6-8: Glossartermdatei mit allen acht Benennungsangaben

Quelle: Codeausschnitt aus XML-Glossartermdatei *term_GT4Tray.xml*

6.3.3 Automatisierungsprozess: Glossar generieren

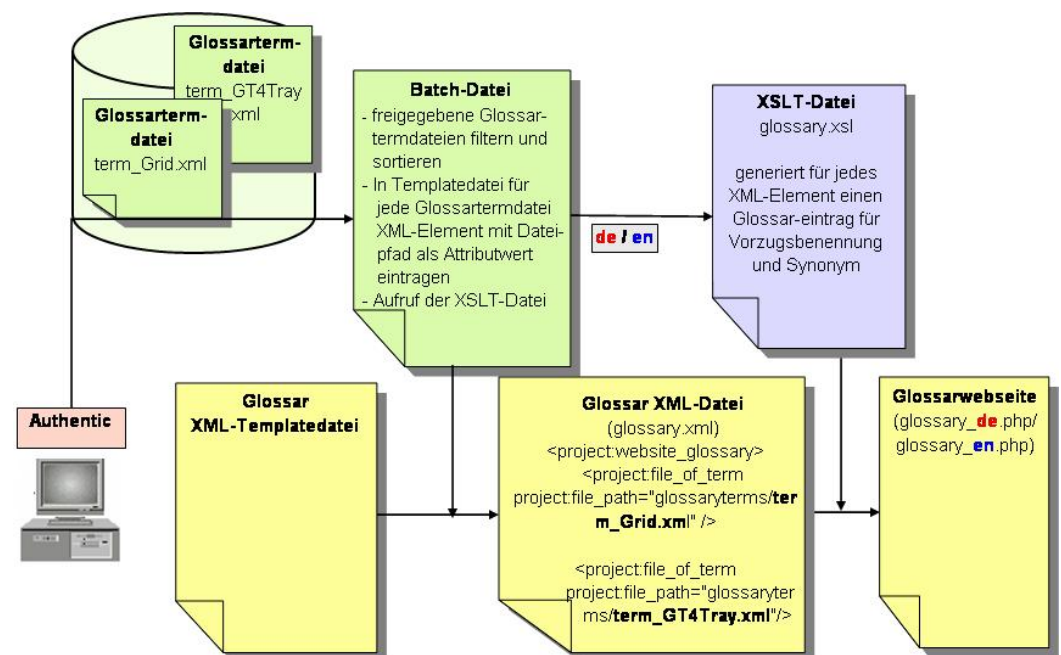
Ziel der
Automatisierung

Das Glossar wird ohne manuellen Eingriff des Users erzeugt.

Zweck der
Automatisierung

Der User kann schnell und unkompliziert ohne Know-how, ein vollständiges Glossar erzeugen, das den aktuellen Stand an freigegebenen Glossartermen wiedergibt. Es entstehen durch den User keine Fehler, da Prozesse wie z.B. das Erfassen von Dateipfaden nicht notwendig sind.

Prozessablauf

Abbildung 6-9: Prozessablauf *Glossar generieren*

Quelle: Eigene Darstellung

In der Rolle des Freigebers initiiert der User per Buttonklick den Prozess. Nicht berechnete Personen erhalten eine entsprechende Meldung. Im aufgehenden

Eingabeformular wählt der User aus der Liste an angebotenen Webseiten die Webseite *Glossar* und die Sprache, in der das Glossar erzeugt werden soll, über Radiobuttons aus (siehe Tabelle 6-3). Zunächst wird die XML-Datei generiert. Im Anschluss wird daraus die Output-Datei mithilfe von XSLT erzeugt. Der Freigeber wird durch Meldungen über den Fortschritt der Generierung und über die Lokalisation des fertig generierten Glossars informiert.

Technische Realisierung

Die Erstellung des Glossars ergibt sich aus dem Zusammenspiel von Skripten im *Authentic*, der XML-Templatedatei sowie der Batch-Datei und XSLT-Datei.

Während *Authentic* das aktuelle Datum und die User-ID der Person übergibt, übernimmt und koordiniert die Batch-Datei den gesamten Generierungsprozess. Wie bereits beim Anlegen eines neuen Projektes werden an die Stelle von Platzhaltern in der XML-Templatedatei Datum und Name eingetragen. Hierfür ist unter anderem eine Trennung des Datums in mehrere Variablen nötig. Alle freigegebenen Glossartermdateien werden in sortierter Reihenfolge in einer Textdatei aufgelistet. Falls keine vorhanden sind, wird eine entsprechende Meldung ausgegeben und der Vorgang wird abgebrochen. Die Glossartermdateien werden durch das entsprechende XML-Element ersetzt und der Dateiname als Attributwert eingetragen. Zudem wird überprüft, ob es sich bei dem Glossarterm um einen neuen handelt. Im Anschluss wird der gesamte Inhalt der Textdatei in die XML-Templatedatei eingefügt, der Schemapfad angepasst und die Glossardatei in den Glossarordner verschoben. In Abhängigkeit von der im *Authentic* angegebenen Sprache (*de/en*), wird mithilfe der XSLT-Datei die XML-Datei in die PHP-Glossardatei transformiert. Der User wird darüber informiert, welche Glossartermdateien im Vergleich zum vorherigen Glossar neu hinzugekommen sind und wo die Datei abgelegt wurde.

6.3.4 Automatisierungsprozess: Formel im TeX-Format in Bild im PNG-Format umwandeln

Ziel der Automatisierung

Eine mit LaTeX erstellte Formel wird in zwei Bilder vom Format *PNG* für den Webaufttritt und für eine PowerPoint Präsentation umgewandelt. Beide haben jeweils eine dem Medium entsprechende standardisierte Schriftgröße.

Zweck der Automatisierung

Der User erhält seine Formel im fertigen Bildformat anstatt sie zuerst in ein PDF umwandeln und anschließend mit einem Bildbearbeitungsprogramm zurechtschneiden zu müssen. Zudem harmonisiert die Schriftgröße der Formel in der Webseite mit dem restlichen Text.

Prozessablauf

Der User löst per Buttonklick den Prozess aus und wählt über Eingabeformulare mit Radiobuttons und Dateidialogen aus, ob alle oder nur bestimmte TeX-Dateien mit Formeln in Bilder konvertiert werden. Tooltips erleichtern ihm die Eingabe. Der Vorgang ist analog zum Prozess *Links auf Glossartermdateien setzen*.

zen. Die Batch-Datei generiert aus den Userangaben und der TeX-Datei das Bild.

Technische Realisierung

Die Konvertierung wird wie bei den anderen Prozessen mithilfe einer Batch-Datei und Skripten im *Authentic* durchgeführt. Neben der Ausgangs-TeX-Datei werden außerdem das Paket *pst-eps* von der Suite *ps-tricks* bei LaTeX benötigt, um die Formel vor der Konvertierung zu kennzeichnen sowie das Bildbearbeitungstool *ImageMagick*, um die Konvertierung von *EPS* in *PNG* vorzunehmen.

Die Batch-Datei erhält von *Authentic* den Projektnamen und die Namen der TeX-Dateien, die bearbeitet werden. Werden mehrere TeX-Dateinamen als String übergeben, trennt die Batch-Datei sie in die einzelnen Dateipfade auf. Sollen alle TeX-Dateien konvertiert werden, filtert die Batch-Datei sie aus dem entsprechenden Projektordner heraus. Jede TeX-Datei für eine Formel basiert auf einer Templatedatei, in der der Autor anstelle des Default-Textes seine Formel setzt:

```
\documentclass[11pt]{scrartcl}
\usepackage{pst-eps}
\begin{document}
$Hier steht die Formel. Nur eine pro LaTeX-Datei$
\end{document}
```

Abbildung 6-10: TeX-Templatedatei zur Formelerstellung

Quelle: Templatedatei *template_equation_AL_SWG.tex*

Ist die Formel bereits in der LaTeX-Datei mithilfe der Angaben `\TeXtoEPS` und `\endTeXtoEPS` gekennzeichnet, kann das Ergebnis in LaTeX nicht visuell überprüft werden. Deshalb werden diese Markierungszeilen von der Batch-Datei vor und hinter der Formel eingefügt. Ist die Formel bereits als Bild vorhanden, aber z. B. geändert worden, werden die Markierungszeilen ersetzt. Anschließend wird die TeX-Datei in das Format *DVI*, dieses in *EPS* und *EPS* in *PNG* umgewandelt und im vorgesehenen Ordner des Projektes gespeichert. Dieser Vorgang wird zuerst für den Webauftritt ausgeführt und im Anschluss für eine PowerPoint Präsentation. Hierzu werden zusätzlich in der TeX-Datei die Angaben zur Schriftgröße und zum Mediumstyp mithilfe des SED-Befehls durch die neuen Angaben ersetzt.

6.3.5 Automatisierungsprozesse: Metadaten eintragen und Anschlussprozesse ausführen

Ziel der Automatisierung

Die Metadaten zum Content-Status einer XML-Datei werden automatisch befüllt (siehe Kapitel 7.4) und die mit ihnen verbundenen Anschlussprozesse ausgeführt.

Zweck der Automatisierung

Zusammen sichern die Automatisierungsmechanismen die Aktualität und Korrektheit derjenigen Metadaten, die den Status einer XML-Datei beschreiben. Der

User bekommt dadurch die Sicherheit, den Dateistatus korrekt vorzufinden. Zudem reduzieren die Prozesse seinen Arbeitsaufwand, da er manuell keine Einträge zum Dateistatus vornehmen muss, sobald dieser sich geändert hat. Außerdem benötigt der User keine Kenntnisse über die mit den Metadaten verbundenen Anschlussprozesse und ihre Besonderheiten. Tooltips und Meldungen leiten den User analog zu anderen automatisierten Prozessen.

Prozessablauf

Automatisierungsprozesse	Prozessablauf
Userdaten abfragen	Diese Automatisierung nimmt einen Sonderstatus ein, da sie für die anderen automatisierten Prozesse die Voraussetzung bildet. ▪ Sobald der User eine Projektdatei im <i>Authentic</i> öffnet, erscheint ein Eingabeformular, in dem er seinen Namen und seine Rolle angibt. Für einen User, der die Rolle des Lektors, Freigebers oder Herausgebers innehat bzw. für das Archivieren zuständig ist, besteht die Möglichkeit, mehrere Rollen anzugeben. So kann er das Dokument nacheinander in seinen verschiedenen Funktionen bearbeiten ohne sich erneut anmelden zu müssen. ▪ Ein User, der eine Rolle angibt, für die er nicht berechtigt ist, erhält eine entsprechende Meldung und der Anmeldevorgang wird wiederholt. ▪ Hat sich der User als Bearbeiter angemeldet, werden beim Betätigen der Speichertaste die Einträge der Metadaten <i>reviewed</i>, <i>approved</i>, <i>published</i> und <i>archived</i> durch die Default-Werte ersetzt, um den Status des Dokumentes zurückzusetzen. ▪ Meldungen informieren den User über den Vorgang.
Datei als <i>created</i> markieren	Der User markiert im <i>Authentic</i> per Buttonklick das Dokument als <i>created</i>. ▪ Default- oder bereits vorhandene Werte der entsprechenden Metadaten werden durch das aktuelle Datum und den Usernamen ersetzt. ▪ Es wird überprüft, zu welchem Webseitentyp die XML-Datei gehört und ggf. ein Übergabeparameter für die anschließende Transformation bereitgestellt. ▪ Die XML-Datei wird in PHP/HTML transformiert, außer es handelt sich um die Metadatendatei eines Bildes. ▪ Die Output-Datei der Transformation wird im vorgesehenen Ordner gespeichert. ▪ Meldungen informieren den User über den Vorgang.
Datei als <i>modified</i> markieren	Der User markiert im <i>Authentic</i> per Buttonklick das Dokument als <i>modified</i>. ▪ Die Default-Werte der entsprechenden Metadaten werden durch das aktuelle Datum und den Usernamen ersetzt. ▪ Meldungen informieren den User über den Vorgang.
Datei als <i>reviewed</i> markieren	Der User markiert im <i>Authentic</i> per Buttonklick das Dokument als <i>reviewed</i>. ▪ Die Default- oder bereits vorhandenen Werte der entsprechenden Metadaten werden durch das aktuelle Datum und den Usernamen ersetzt. ▪ Meldungen informieren den User über den Vorgang.
Datei als <i>approved</i> markieren	Der User markiert im <i>Authentic</i> per Buttonklick das Dokument als <i>approved</i>. ▪ Die Default- oder bereits vorhandenen Werte der entsprechenden Metadaten werden durch das aktuelle Datum und den Usernamen ersetzt. ▪ Der Dateipfad der Output-Datei wird im vorgesehenen Attribut des Metadatums <i>outputfile</i> eingetragen. ▪ Es wird überprüft, zu welchem Webseitentyp die XML-Datei gehört und ggf. ein Übergabeparameter für die anschließende Transformati-

Automatisierungsprozesse	Prozessablauf
	on bereitgestellt. - Die XML-Datei wird in PHP/HTML transformiert, außer es handelt sich um die Metadatendatei eines Bildes. - Die Output-Datei der Transformation bzw. das Bild der Metadatendatei werden im vorgesehenen Ordner gespeichert. - Meldungen informieren den User über den Vorgang.
Datei als <i>published</i> markieren	Der User markiert im <i>Authentic</i> per Buttonklick das Dokument als <i>published</i>. - Die Default-Werte der entsprechenden Metadaten werden durch das aktuelle Datum und den Usernamen ersetzt. - Die Versionsnummer wird im entsprechenden Metadatum um eins erhöht. - Es wird überprüft, ob es sich um die XML-Datei des Webseitentyps <i>overview_projcet</i> handelt und bei Zutreffen wird der Prozess <i>Glossar-term zum Projektnamen befüllen</i> angestoßen (siehe Tabelle 6-3). - Es wird überprüft, ob in die XML-Datei Bilder eingebunden sind. Trifft dies zu, werden in den Metadatendateien der Bilder die Default-Werte des Metadatum <i>published</i> durch das aktuelle Datum und den Usernamen ersetzt. - Meldungen informieren den User über den Vorgang.
Datei als <i>archived</i> markieren	Der User markiert im <i>Authentic</i> per Buttonklick das Dokument als <i>archived</i>. - Die Default-Werte der entsprechenden Metadaten werden durch das aktuelle Datum und den Usernamen ersetzt. - Es wird überprüft, ob in die XML-Datei Bilder eingebunden sind. Trifft dies zu, werden in den Metadatendateien der Bilder die Default-Werte des Metadatum <i>archived</i> durch das aktuelle Datum und den Usernamen ersetzt. - Anlegen eines Tags: Für den gesamten Output-Ordner <i>web</i> wird im Versionierungssystem <i>SVN</i> ein Verzeichnis angelegt, das auf alle aktuellen Versionen der Dateien des Ordners referenziert. - Meldungen informieren den User über den Vorgang.

Tabelle 6-2: Prozessabläufe zum Eintragen von Metadaten

Quelle: Eigene Darstellung

Um die Anzahl an Buttons für den User übersichtlich zu halten, steht nur *ein* Button für die in Tabelle 6-2 beschriebenen Prozesse zur Verfügung. Je nach angegebener Rolle im Eingabeformular werden die entsprechenden Metadaten befüllt, sobald er betätigt wird.

6.3.6 Weitere Automatisierungsprozesse

Einen Überblick zu den Abläufen weiterer Automatisierungsprozesse zeigt folgende Tabelle:

Automatisierungsprozesse	Prozessablauf
Webseite auswählen	Diese Automatisierung nimmt einen Sonderstatus ein, da sie für die anderen Prozesse, die automatisch Webseiten generieren oder XML-Dateien mithilfe von Templatedateien anlegen, die Voraussetzung bildet: - Der User startet per Buttonklick den Prozess. - Ein Eingabeformular mit einer Auflistung an Webseiten bzw. XML-Dateien erscheint, die über Radiobuttons ausgewählt werden können:

Automatisierungsprozesse	Prozessablauf
	Glossar, Aktuelle Projekte, Alte Projekte, Glossarterm, Metadatendatei für Bild, Metadatendatei für den gesamten Webauftritt, alle Dateien in PHP/HTML transformieren. Für mehrere Webseiten bzw. Dateien muss zusätzlich die Sprache der Outputdatei über Radiobuttons angegeben werden. Entsprechend der Angaben wird der jeweilige Automatisierungsprozess aufgerufen und an ihn Parameter übergeben (Sprache/ Webseiten-ID).
Metadatendatei für neue Bilder anlegen	Der Prozess wird durch den Vorgängerprozess <i>Webseite auswählen</i> ausgelöst, von dem er die Sprache der Webseite erhält, in dem das Bild erscheint. - Voraussetzung: Prozess <i>Projekt generieren</i> (siehe Kapitel 6.3.1). Dieser Prozess legt für das Projekt die angepasste Metadaten-Template-datei (<i>metadata_imagename.xml</i>) im entsprechenden Unterordner des Projektordners an. Für projektunabhängige Bilder ist die Metadaten-Templatedatei fest vorhanden. - Eine Meldung informiert den User über den Vorgang. - Es wird mithilfe der Vorlagendatei für jedes neue Bild eine Datei angelegt und der Dateiname entsprechend angepasst. - Der Dateipfad des Bildes wird in der Metadatendatei anstelle des Platzhalters als Attributwert des Elementes <i>image</i> eingefügt. - Die Dateipfade zu den Metadatendateien werden in der <i>Authentic-Projektdatei</i> eingetragen. - Die Dateien erscheinen in der Ordnerstruktur des geöffneten Projektes im <i>Authentic</i>. - Eine Meldung informiert den User über das Prozessende.
Reminder, für ein neues Bild eine Metadatendatei erzeugen	Der User hat sich als Autor eingeloggt (Information stammt aus Prozess <i>Userdaten abfragen</i>). - Beim Autor eines Projektes wird überprüft, ob sich in seinem Projektordner für Bilder Photos befinden, für die noch keine Metadatendateien vorliegen. - Beim Verantwortlichen für projektunabhängige Webseiten wird dies analog für projektunabhängige Bilder durchgeführt (z.B. für die Webseiten <i>Glossarterm</i>, <i>Ziele</i>). - Der Autor wird per Meldung informiert, wenn ein solches Bild gefunden wurde. - Entscheidet sich der Autor bzw. Verantwortliche durch Betätigen der OK-Taste dafür, die entsprechenden Metadatendateien anzulegen, wird der Prozess <i>Metadatendatei für neue Bilder anlegen</i> aufgerufen. - Entscheidet er sich dagegen, wird der Prozess beendet.
Glossartermdatei des Projektnamens füllen	Der Vorgängerprozess <i>Datei als published markieren</i> löst den Prozess aus (siehe Tabelle 6-2). - Voraussetzung: Die Glossartermdatei für den Projektnamen ist aus der entsprechenden Templatedatei (<i>term_projectname.xml</i>) generiert und im projektübergreifenden Ordner für Glossartermdateien vorhanden. Sie enthält bereits den Autornamen und ist in der <i>Authentic-Projektdatei</i> aufgeführt. Alle Punkte werden vom Prozess <i>Projekt generieren</i> beim Anlegen des Projektes übernommen. - Eine Meldung informiert den User über den Vorgang. - Die Kurzbeschreibung aus der Übersichtsseite ersetzt den Platzhalter in der Glossartermdatei. - Die Glossartermdatei wird in der Ordnerstruktur des geöffneten Projektes im <i>Authentic</i> angezeigt und öffnet sich im Editor. - Der User wird per Meldung aufgefordert, weitere Benennungen für den Projektbegriff anzugeben, falls welche existieren.
Glossarterm anlegen	- Der Vorgängerprozess <i>Webseite auswählen</i> löst den Prozess aus - Ein Eingabeformular erscheint, in das der User die Vorzugsbenennung des Glossarbegriffes einträgt.

Automatisierungsprozesse	Prozessablauf
	▪ Mithilfe der XML-Templatedatei (<i>term_termname.xml</i>) wird eine neue Glossartermdatei angelegt und in den Ordner für Glossarterme kopiert. ▪ Der Dateiname wird entsprechend der Vorzugsbenennung angepasst. ▪ Der Username, den der Prozess <i>Userdaten abfragen</i> bereitstellt (siehe Tabelle 6-2), wird in der Datei anstelle der Default-Werte des Metadatums <i>created</i> eingetragen. ▪ Auch der Schemapfad wird angepasst. ▪ Die Datei erhält in der <i>Authentic</i>-Projektdatei einen Eintrag mit Angabe des Dateipfades. ▪ Die Glossartermdatei wird in der Ordnerstruktur des geöffneten Projektes im <i>Authentic</i> angezeigt und öffnet sich im Editor. ▪ Meldungen informieren den User über den Vorgang.
Webseitentypen <i>Aktuelle Projekte</i> oder <i>Alte Projekte</i> automatisch generieren	Der Vorgängerprozess <i>Webseite auswählen</i> löst den Prozess aus, der die Webseiten-ID und die Sprache der gewünschten Webseite übergibt. ▪ Eine Meldung informiert den User über den Vorgang. ▪ Mithilfe der XML-Templatedatei (<i>projects_en.xml</i>), die für die jeweilige Sprache vorgesehen ist, wird eine neue XML-Webseitendatei angelegt. ▪ Die alte Webseitendatei wird durch die neue im Ordner <i>projects</i> ersetzt. ▪ Jedes Projekt erhält in der Datei entsprechend den Schemavorgaben einen Eintrag mit Angabe des Dateipfades zur Übersichtsseite als Attributwert. ▪ Der Schemapfad wird angepasst. ▪ Die Transformation der Datei wird aufgerufen und der XSLT Datei die Webseiten-ID übergeben (siehe Kapitel 9.5.2). ▪ Die generierte Outputdatei wird im entsprechenden Ordner gespeichert. ▪ Eine Meldung informiert den User über das Prozessende.
Alle XML-Dateien in PHP/HTML umwandeln	Der User startet den Prozess per Buttonklick. ▪ Alle XML-Webseitendateien des im <i>Authentic</i> geöffneten Projektes werden mithilfe der entsprechenden XSLT-Stylesheets in PHP/HTML umgewandelt. ▪ Die Outputdateien werden in die entsprechenden Ordner verschoben. ▪ Meldungen informieren den User über den Vorgang.
Metadatendatei für gesamten Webauftritt generieren	Der Vorgängerprozess <i>Webseite auswählen</i> löst den Prozess aus. ▪ Eine Meldung informiert den User über den Vorgang. ▪ In der XML-Templatedatei (<i>overview_metadata_projects.xml</i>) werden Username und Datum anstelle der Platzhalter des Metadatums <i>created</i> eingetragen. ▪ Alle XML-Dateien des Ordners <i>src</i> mit Ausnahme der Dateien des Unterordners <i>share</i> erhalten in der Templatedatei entsprechend den Schemavorgaben einen Eintrag. ▪ Der Schemapfad wird angepasst und die XML-Datei in den vorgesehenen Ordner verschoben. ▪ Die Datei wird mithilfe von XSLT in HTML umgewandelt und die Outputdatei im entsprechenden Ordner gespeichert. ▪ Eine Meldung informiert den User über das Prozessende.

Tabelle 6-3: Ablauf weiterer Automatisierungsprozesse

Quelle: Eigene Darstellung

7 Ablagekonzept und seine Realisierung

7.1 Einführung

Das Ablagekonzept besteht aus der Ordnerstruktur sowie den Namens- und Metadatenregelungen für die einzelnen Dateien.

7.2 Vorgehensweise bei der Konzeption

Kriterien für die Ordnerstruktur

Es wurden Kriterien aufgestellt, die die Ordnerstruktur erfüllen soll:

- Schnelles Auffinden der gewünschten Informationen mithilfe kurzer Navigationswege und intuitiv verständlicher Ordner- und Dateinamen
- Keine Redundanz bei der Datenhaltung
- Leicht erweiterbare Ordnerstruktur
- Identische relative Dateipfade im Erstellungsordner und bei der Veröffentlichung auf dem Webserver

Usability-Tests

Während der Konzeptionsphase wurden die Zwischenstände der Ordnerstruktur anhand von Usability-Tests auf ihre Verständlichkeit hin überprüft. Bei diesen Tests wurde eine Kombination aus *Wording-Test*, *Card-Sorting* in abgewandelter Form und *Usability Testing* eingesetzt (siehe CD Datei *Usability_Fragebogen.doc*). Auf Grundlage der Ergebnisse wurde die Ordnerstruktur verbessert. Das Szenario - ohne die Hilfe eines Redaktionsleitfadens zu navigieren - entspricht dem typischen Nutzerverhalten in der Grid-Softwaregruppe.

Dateinamenregelungen

Standardisierung bei den Dateibenennungen bildet die Grundlage für die Erstellung der Webseiten mit XSLT, aber auch für die Automatisierungsprozesse. Deshalb wurden Regeln zur Dateibenennung aufgestellt und anhand von Beispielen verdeutlicht. Es wurde darauf geachtet, bei den Benennungsschemas Erweiterungsmöglichkeiten zu berücksichtigen (Ergebnis siehe CD Datei *Ablagekonzept.xls*).

Metadateneinschränkung bei viewCVS und TortoiseSVN

Damit die Grid-Softwaregruppe den Überblick zum Stand des Webauftrittes bzw. zu ihren einzelnen Dateien behält, mussten Metadaten zu den einzelnen Dateien festgelegt werden. In SVN können zwar zu den Default-Metadaten eigene Metadaten angelegt werden, in der Webansicht mithilfe von viewCVS werden diese aber nicht angezeigt. Zudem werden die Angaben beim Einchecken leicht vergessen. Bei TortoiseSVN kann sich der User zwar alle Metadaten anzeigen lassen, aber nur pro Datei. Das Auslesen aller in TortoiseSVN angegebenen Metadaten mithilfe einer Batch-Datei erwies sich als unpraktikabel. Für den Überblick zu den Metadaten wurde deshalb nach einer anderen Lösung gesucht. Beim Aufstellen geeigneter Metadaten galt der Grundsatz „Weniger ist mehr“, damit die

definierten Metadaten von der Grid-Softwaregruppe gepflegt werden. Die in Kapitel 6.3 aufgeführten Automatisierungsprozesse leisten dazu einen wichtigen Beitrag. Auch wurde auf selbsterklärende XML-Element- und Attributnamen geachtet.

7.3 Ergebnis der Ordnerstruktur

Der Gesamtordner *GRID_projects* und der Ordner für ein Projekt weisen folgende Struktur auf:

Überblick Gesamt- und Projektordnerstruktur

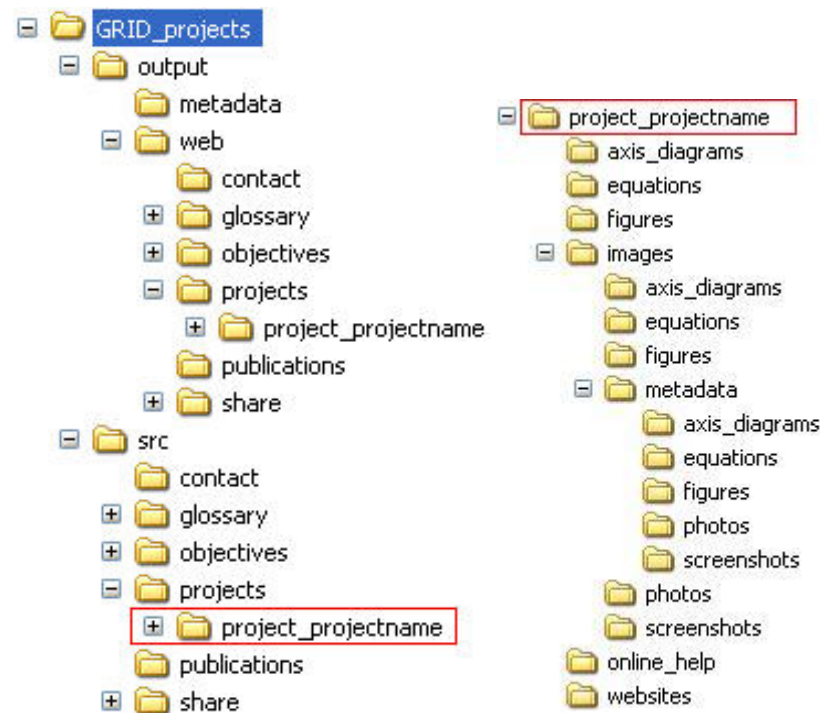


Abbildung 7-1: Ordnerstruktur von *GRID_projects* und eines unbenannten Projektordners

Quelle: Eigene Darstellung

Jedem Unterordner von *src* entspricht ein Unterordner von *web*, damit sich der User leicht zurecht findet und die relativen Pfade der Webseitendateien und Bilder zueinander erhalten bleiben und nicht über XSLT geändert werden müssen. Der Ordner *web* bietet zudem die Möglichkeit, den Fortschritt des Webauftrittes leicht zu kontrollieren, da er nur freigegebene Dateien enthält. Auch vereinfacht er die Veröffentlichung und Archivierung des aktuellen Webauftrittes, indem er auf den Webserver kopiert bzw. in SVN zusätzlich referenziert wird. Einziger Nachteil besteht in der doppelten Datenhaltung bezüglich der projektspezifischen und gemeinsamen Bilddateien sowie dem CSS-Stylesheet und den JavaScript-Dateien. Um die Aktualität der Bilder im Ordner *web* dennoch zu gewährleisten, existieren Automatisierungsprozesse, die den Abgleich automatisch durchführen (siehe Tabelle 6-2). Die Ordner sind nach den Webseitentypen benannt, deren Dateien sie beinhalten. Neben dem Ordner *metadata* bildet noch der schreibgeschützte Ordner *share* eine Ausnahme. Zugriffsrecht hat nur der Informationsde-

signer, da er in seinen Unterordnern alle Dateien enthält, die für die Webseiten und Automatisierungsprozesse die Basis bilden. Dazu gehören z.B. Skripte, XML-Templatedateien, Schemas, Stylesheets und gemeinsame Bilder (siehe Anhang E). Zudem wurde die Ordnerstruktur so konzipiert, dass Erweiterungen möglich sind:

Erweiterbarkeit der Ordnerstruktur

Erweiterungsmöglichkeit	Bisher	Künftig
Medientyp	Ordner <i>output</i> mit Ordner <i>web</i>	Ordner <i>output</i> mit Ordner <i>pdf</i>

Tabelle 7-1: Erweiterungsmöglichkeit der Ordnerstruktur

Quelle: Eigene Darstellung

Die Startseite des gesamten Webauftrittes befindet sich direkt im Ordner *src* bzw. *web*; die Startseite eines Projektes direkt im Projektordner, in dem auch die Projektdatei für den Editor *Authentic* abgelegt ist.

Vorteile der Projektordnergliederung

Ausschlaggebend für die Gliederung des Projektordners war, dass sie die aufgestellten Kriterien erfüllt:

- *Keine mehrfache Datenhaltung*: Alle Webseiten eines Projektes nutzen denselben Datenpool (Formeln, Zeichnungen etc.).
- Nach Änderung eines Bildes *kurze Aktualisierungsdauer*: Alle Webseiten, die das Element verwenden, sind sofort aktualisiert.
- *Kompakt*: Der Nutzer erhält alle seine Instanzen eines Elementtyps auf einen Blick.
- *Kurze Navigationswege* durch flache Hierarchien innerhalb eines Projektes.
- *Verständliche Ordnernamen*, da sie aus dem Projektumfeld der Grid-Softwaregruppe stammen.

Sonderfall: Software und Publikationen

Publikationen- und Software sind nicht in die Ordnerstruktur integriert, da sie auf Wunsch des Technologie-Beauftragten der Grid-Softwaregruppe wegen der großen Datenmenge in Ordnern auf einem anderen Server liegen. Hierfür wurde eine weitere Ordnerstruktur angelegt (siehe Anhang E).

7.4 Ergebnis der Metadatenregelungen für Dateien

Für die Metadaten der Dateien wurde festgelegt, welche Dateien durch welche Metadaten gekennzeichnet sind, wann und wie die Metadaten gesetzt werden und wie der User die Metadaten aller Webseiten kompakt abfragen kann (siehe CD Datei *Ablagekonzept.xls*).

Jede Datei, die sich im Ordner *GRID_projects* befindet, erhält die drei von *TortoiseSVN* automatisch gesetzten Metadaten: Name der Person, die die Datei ein-

gecheckt hat, Datum des Eincheckvorganges und Revisionsnummer. Zudem gibt der User beim Einchecken über das Metadatum *Kommentar* die vorgenommenen Änderungen an. Dazu wählt er aus einer Zahl an Standardkommentaren, die von der Autorin definiert wurden, den passenden aus. Die XML-Webseitendateien enthalten zusätzlich zur genaueren Spezifizierung einen Metadatenabschnitt mit Angaben zum Content-Status (*content_status*), zu den Dokumenten- (document_data) und – bei Projektseiten – zu den Projektrahmendaten (*project_data*). Deren Kindelemente sind je nach Funktion erforderlich oder optional.

```
<project:document_data>
  <project:language>en</project:language>
  <project:version>0.0</project:version>
  <project:overview_of_keywords>
    Automatisierungsprozesse, Projektstruktur für Softwareprojekte,
    Technologie-Architektur, Standardisierung
  </project:overview_of_keywords>
  <project:overview_of_output>
    <project:outputfile project:format="html" project:file_path="" />
  </project:overview_of_output>
</project:document_data>
```

Abbildung 7-2: Metadatenabschnitt *document_data* in einer XML-Webseitendatei

Quelle: Ausschnitt aus Beispiel-Webseite

Im Gegensatz zur Revisionsnummer von *TortoiseSVN*, die die Anzahl an Eincheckvorgängen mit Dateiänderung angibt, steht die Versionsnummer im Metadatenabschnitt für die Anzahl der Veröffentlichungen.

Wird eine XML-Webseite automatisch erzeugt, erhält sie eine verkürzte Version der Metadaten aus dem Bereich *content_status*: *approved*, *published* und *archived*. Die Befüllung der XML-Metadaten mit Werten erfolgt auf folgende Weise:

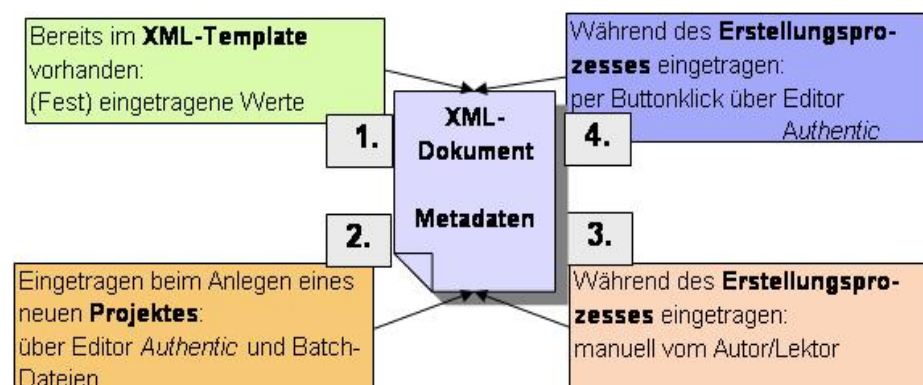


Abbildung 7-3: Befüllungsmethoden der Metadaten einer XML-Datei

Quelle: Eigene Darstellung

XML-Metadatendatei
für ein Webseitenbild

Für jedes Webseitenbild gibt es eine eigene XML-Metadatendatei (siehe Tabelle 6-3), um auch zu ihm dieselben Informationen zu erhalten wie zu den XML-

Webseiten. Es wird unterschieden, ob das Bild zu einem Projekt gehört oder nicht. Das XML-Dokument enthält neben dem Metadatenabschnitt noch den Dateipfad, so dass der User im *Authentic* das Bild anschauen kann und je nach Rolle die Metadaten eingetragen werden (siehe Tabelle 6-2).

HTML-Überblicksseite
zu den Metadaten

Mithilfe einer eigenen Webseite werden die in den XML-Webseiten und Bilddateien festgehaltenen Metadaten dem User übersichtlich und kompakt angezeigt. Sie wird vom User per Buttonklick erzeugt (siehe Tabelle 6-3).

8 Webauftrittskonzept

8.1 Einführung

Das Konzept für den Webauftritt umfasst:

- die Webauftrittsstruktur und ihre Webseitentypen mit Metadaten (Siehe Kapitel 8.3)
- die Beziehungen der Webseiten untereinander und nach außen (Siehe Kapitel 8.4)

8.2 Problematische Rahmenbedingungen

Gegen Ende der technischen Realisierungsphase des ausgearbeiteten Konzeptes wurden die Webseiten des Forschungszentrums und des Instituts für Prozessdatenverarbeitung und Elektronik (IPE) grundlegenden Änderungen unterzogen und damit eine komplett neue Ausgangssituation geschaffen (siehe Anhang F). Ehemals elementare und einschränkende Rahmenbedingungen in den Gestaltungsmöglichkeiten entfielen, neue entstanden:

Änderung der Rahmenbedingungen und ihre Auswirkungen

Änderungskategorie	Alter Webauftritt	Konzeptanforderungen alter Webauftritt	Neuer Webauftritt
Seitenaufbau und Zielfenster des Webauftrittes	▪ Frames strukturieren die FZK-Webseiten ▪ Webauftritt öffnet sich im Frame der aktuellen Contentseite	Grid-Softwaregruppe benötigt Webseiten zum Einbinden in den Frame	Viele ineinander geschachtelte unsichtbare Tabellen strukturieren die FZK-Webseiten
Vorhandene Navigation	Der Webauftritt ist durch die anderen beiden Frames von drei Navigationsmenüs umgeben	▪ Um den User durch die Navigationsvielfalt nicht zu verwirren, darf höchstens eine weitere Navigationsleiste eingesetzt werden, die statisch ist. ▪ Für das Navigieren in tiefere Ebenen muss eine andere Lösung gefunden werden	▪ Webauftritt ist durch das Öffnen in einem Zweitfenster nicht mehr durch Navigationsmenüs eingeschränkt. ▪ Dynamische Pull-down-Menüs sind möglich.
Zugriff auf Startseite	Der Zugriff ist über Eintrag <i>Grid</i> im linken Navigationsmenü (Frame) stets gewährleistet.	Keine Notwendigkeit den Zugriff auf die Startseite durch einen weiteren Eintrag im Navigationsmenü des Webauftrittes anzubieten	Zugriff ist nur noch über Link <i>Wissenschaftliche Software</i> im Erstfenster möglich.

Tabelle 8-1: Gegensatz der Ausgangsbedingungen alter/neuer IPE-Webauftritt

Quelle: Eigene Darstellung

Der von der Autorin konzipierte und umgesetzte Webauftritt basiert auf den Vorgaben des alten IPE-Webauftrittes. Der zeitliche Rahmen der Master-Thesis erlaubte es nicht, diesen an die neuen Gegebenheiten anzupassen.

Zurzeit sind sowohl der alte als auch der neue Webauftritt des Institutes im WWW erreichbar. Der User findet den alten Webauftritt weiterhin unter der URL <http://www.ipe.fzk.de>, den neuen über die Startseite des Forschungszentrums: <http://www.fzk.de/fzk/idcplg?IdcService=FZK&node=4605>.

8.3 Webauftrittstruktur und Webseitenkonzept

8.3.1 Vorgehensweise bei der Konzeption

Informationsarchitektur bedeutet methodisches und strukturelles Informationsdesign, also Design, um aus einer diffusen Menge von Daten „Informationen“ zu machen. (Manhartsberger, Musil Web Usability:110)

Bei der Vorgehensweise zur Webauftrittskonzeption orientierte sich die Autorin an den Empfehlungen aus der Fachliteratur und Normen für userfreundliche Internetauftritte, die in Kapitel 3.11 aufgeführt sind.

Zielgruppendefinition	Zunächst wurden die Zielgruppen definiert. Dabei wurden zwei Sichtweisen eingenommen: Wer soll aus Sicht der Softwaregruppe angesprochen werden und wer kann im Hinblick auf die Softwareprodukte Zielgruppe sein und den Webauftritt nutzen. Anhand einer Matrix wurden die Webseitennutzer charakterisiert und gruppiert.
Inhaltseingrenzung	Um herauszufinden, welche Inhalte für den Webauftritt relevant und welche Vorkenntnisse dazu nötig sind, wurden ebenfalls zwei Sichtweisen eingenommen: die Sicht des Anbieters und die Sicht der Zielgruppen, um so einen gemeinsamen Nenner zu finden. Die Anforderungen bzw. Wünsche seitens der Grid-Softwaregruppe an den Webauftritt wurden geklärt. Ausgangspunkt bildeten dabei die gewonnenen Erkenntnisse aus der Ist-Analyse (siehe Kapitel 4.3) und der Schwachstellenanalyse des aktuellen Webauftrittes.
Schwachstellen des Webauftrittes	Eine kritische Überprüfung des bisherigen Webauftrittes hinsichtlich der unter Kapitel 3.11 erwähnten Kriterien Navigation, Seitenaufbau, Inhalt, Layout, Browserdarstellung und Seitenidentifizierung im WWW, zeigte folgende Schwachstellen: ▪ Eine Basisnavigation, die jederzeit sichtbar ist, fehlt. ▪ Es existieren „tote“ Links. ▪ Die Seiten weisen keine einheitliche Seitenstruktur auf. Der Seitenaufbau ist weder mediengerecht noch nutzerfreundlich gestaltet: Die Seiten enthalten zum Teil zu viel Inhalt, so dass der User mehrmals scrollen muss. Zudem sieht er sich leserunfreundlichen Zeilenlängen und auf einigen

Webseiten endlosen Fließtextpassagen gegenüber. Auch aussagekräftige Webseitentitel fehlen.

- Viele Projekte sind nicht durch einen Webauftritt vertreten, speziell die Projekte von Studenten fehlen. Die vorhandenen Projekte sind mit Ausnahme des Projektes *GT4Tray* mit jeweils einer Webseite vorhanden. Sie unterscheiden sich drastisch im Informationsgehalt.
- Die Inhalte sind zum Teil veraltet, eine kontinuierliche Pflege der Webseiten ist nicht vorhanden.
- Ein einheitliches Layout fehlt. Farbgestaltung, Schriftgröße und Schriftart unterscheiden sich von Seite zu Seite. Auch weisen manche Webseiten drei bis vier unterschiedliche Schriftgrößen auf.
- Die Webseiten sind nicht konform mit der Corporate Identity des Forschungszentrums. Auf manchen Seiten wirkt dies disharmonisch mit den Frameinhalten, die über das CMS ausgegeben werden und damit der Corporate Identity des Forschungszentrums unterliegen.
- Die Farben der Webseiten werden von den Browsern zum Teil unterschiedlich dargestellt.
- Auf den Gebrauch von Meta-Tags wird verzichtet und damit auf die Möglichkeit, eine hohe Trefferquote für die Webseite über Suchmaschinen zu erhalten.

Da die Mitarbeiter und Studenten der Softwaregruppe nicht nur Autoren des Webauftrittes sind, sondern gleichzeitig auch Vertreter einiger Zielgruppen, gab ihre Befragung Antwort darauf, welche Inhalte die Zielgruppen zu den einzelnen Softwareprojekten und -produkten in einem Webauftritt erwarten. Vorhandene Webauftritte wurden zudem auf die Funktionen der Inhaltsbausteine hin untersucht, um für die in der Matrix beschriebenen Zielgruppen wichtige Inhaltsmodule zu identifizieren. Zusätzlich gaben Fragen seitens der Zuhörer bei Institutsvorträgen Aufschluss darüber, welche Informationen die Zielgruppen interessieren und welches Vorwissen vorhanden sein muss. Diese Möglichkeit bot auch die *German e-Science Conference 2007*.

Grobkonzept

Danach wurde ein Grobkonzept entwickelt und die Webseitentypen zum Teil mit dem Tool *Dreamweaver* umgesetzt. Auf diese Weise wurden der Abteilungsleitung Vorschläge unterbreitet und überarbeitet.

Feinkonzept Für jeden Webseitentyp wurde unter Beachtung von Usability-Gesichtspunkten (siehe Kapitel 3.11) ein Feinkonzept mit technischer Realisierung definiert und nach festgelegten Kriterien in Tabellenform beschrieben (siehe Anhang G).

**Einteilungskriterien
Webseitenmatrix**

Diese Kriterien umfassen Daten ...

- zur Webseitenidentifizierung (z. B. Webseiten-ID, Dateibenennungsschema und PHP/HTML-Beispieldatei)
- zur Positionierung der Webseite im Webauftritt und ihren erlaubten Beziehungsklassen (Linktypen)
- zur Webseitenfunktion und -struktur
- zum Entstehungsprozess der Webseite mit seinen Automatisierungselementen
- zu Besonderheiten der Webseite, die der Autor oder Seitenverantwortliche beachten muss
- zur technischen Realisierung wie
 - Layout (CSS- und SPS-Stylesheet)
 - Zusammenspiel von XSLT-Stylesheets
 - Funktionen
 - globale und lokale Variablen
 - globale und lokale Parameter
 - JavaScript-Skripte

Die Tabelle bildet die Basis für die technische Umsetzung der Webseitentypen, aber auch eine Quelle für deskriptive und prozedurale Webseitenkapitel im Redaktionsleitfaden (siehe Kapitel 10). Je nach Fragestellung des Nutzers sind unterschiedliche Sichtweisen auf die jeweilige Tabelle möglich. Die Abteilung kann so schnell und gezielt auf die verschiedensten Informationen zu einer Webseite zugreifen. Eine aufwändige Suche nach Informationen entfällt dank des zentralen Informationsablagensystems. Gerade im Hinblick auf Änderungen oder Erweiterungen des Webauftrittskonzeptes bietet es eine Möglichkeit, dieses weiterhin mithilfe der Tabelle in standardisierter Form zu dokumentieren.

Zeitlich bedingt konnte im Rahmen der Master-Thesis eine abschließende Überprüfung des Webauftrittes durch Usability Testing nicht mehr durchgeführt werden. Während der Konzeptions- und Realisierungsphase wurden aber die Zwischenergebnisse durch Einbezug der Mitarbeiter als potentielle User des Webauftrittes überprüft.

8.3.2 Ergebnis des Webauftritt- und Webseitenkonzeptes

8.3.2.1 Zielgruppen

Für den Webauftritt der Grid-Softwaregruppe sind folgende Zielgruppen (Nutzergruppen) relevant:

Nutzergruppe	Software-user	Administratoren	Entwickler	Interessierte	Geschäftsführung	Investoren
Vorwissen-kategorie						
Anfänger				I,G	I,G	I,G
Fortgeschrittene	I, G	G	I,G	I		
Experten	I, G	I,G	I,G			

I = Informatik (Kenntnisse in Software, Hardware)

G = Grid-Computing

Tabelle 8-2: Matrix zur Zielgruppenbeschreibung

Quelle: Eigene Darstellung

Die Zielgruppen werden hinsichtlich ihres Vorwissens in drei Kategorien unterteilt: *Anfänger*, *Fortgeschrittene* und *Experten*. Zusätzlich wird bei dem Vorwissen unterschieden, auf welchem Gebiet Kenntnisse vorhanden sind: im Informatikbereich und/oder im Grid-Computing.

8.3.2.2 Sprachen des Webauftrittes

Sprachen des Webauftrittes

Der Webauftritt wurde zunächst in englischer und deutscher Sprache angedacht. Gerade Webseitentypen mit Marketing- bzw. Werbecharakter, die sich auf den obersten Ebenen befinden, sollten auch auf Deutsch vorhanden sein, um Neulingen in punkto *Grid-Computing* den Einstieg in das Thema und in die Softwareprojekte zu erleichtern. In Übereinkunft mit der Gruppenleitung wird das Webauftrittskonzept vorerst nur für die englische Version umgesetzt. Um der Softwaregruppe die Möglichkeit zu geben, in der Zukunft auch einen deutschen Webauftritt umzusetzen, wurde dies in der Konzeption und technischen Realisierung berücksichtigt und umgesetzt.

8.3.2.3 Webauftrittstruktur

Struktur des gesamten Webauftrittes

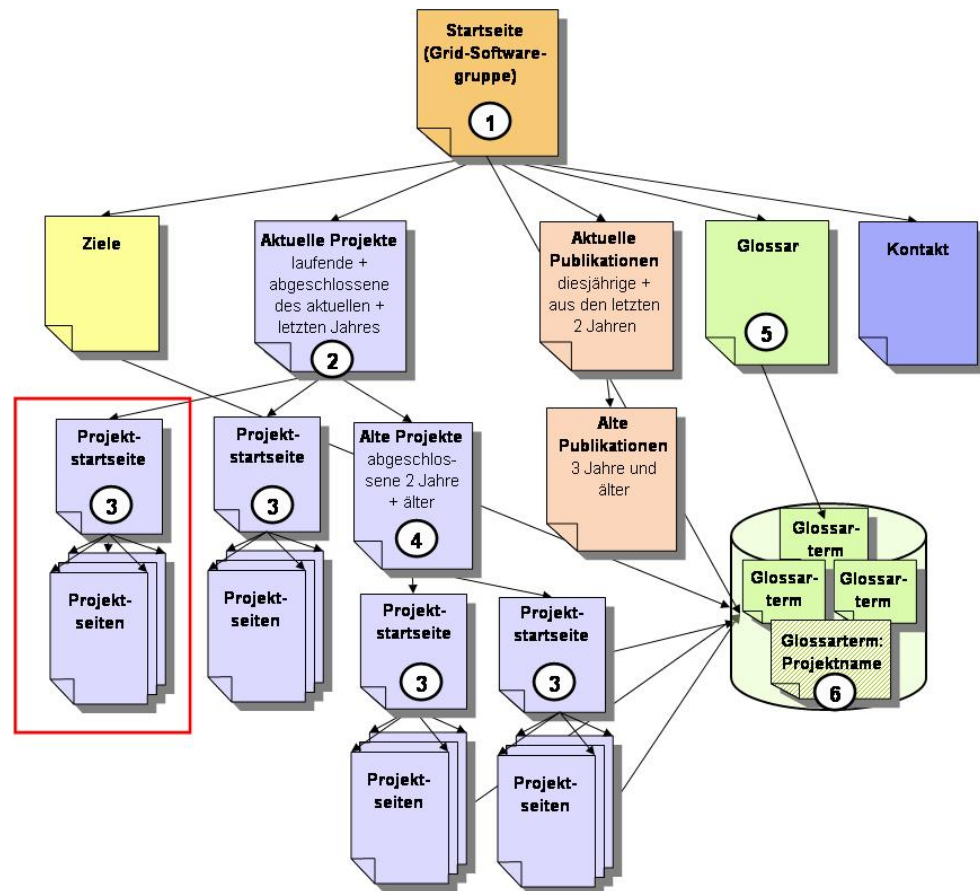


Abbildung 8-1: Übersicht zum gesamten Webauftritt

Quelle: Eigene Darstellung

Über den IPE-Webauftritt erreicht der User die Startseite der Grid-Softwaregruppe. Von dort kann er auf die Webseiten der obersten Ebene *Ziele*, *Aktuelle Projekte*, *Aktuelle Publikationen*, *Glossar* und *Kontakt* zugreifen. Die projektunabhängigen Webseiten beinhalten Themen, die dem User einen Einblick in die Arbeit der Grid-Softwaregruppe geben. Unterhalb der Webseite *Aktuelle Projekte* sind die Webauftritte aller laufenden und abgeschlossenen Projekte des aktuellen und letzten Jahres angesiedelt. Bei den laufenden Projekten erreicht der User die *Projektüberblicksseite* in ihrer Kurzfassung, bei den abgeschlossenen die *Projektstartseite* (siehe Kapitel 8.3.2.4). Abgeschlossene Projekte, die älter als ein Jahr sind, befinden sich unterhalb der Webseite *Alte Projekte*, die wiederum über die Webseite *Aktuelle Projekte* erreichbar ist. Analog zu den Projekten werden die Publikationen in *Aktuelle* und *Alte Publikationen* unterteilt und mit je einer Webseite umgesetzt. Auch hier ist die Webseite für alte Publikationen über die Webseite der aktuellen erreichbar. Nur die projektunabhängigen Webseiten *Startseite*, *Ziele* und *Glossar* können Glossarterme enthalten und auf die entsprechenden Glossarterm-Webseiten verlinken.

Automatisiert erstellte
Webseiten

Die Ziffern 1 - 6 in Abbildung 8-1 symbolisieren Webseiten, die nicht manuell erstellt werden, sondern automatisiert. Der User erzeugt per Buttonklick die fertige Outputdatei (siehe Kapitel 6.3.3, 6.3.6, 9.5).

Struktur eines
Projekt-Webauftrittes

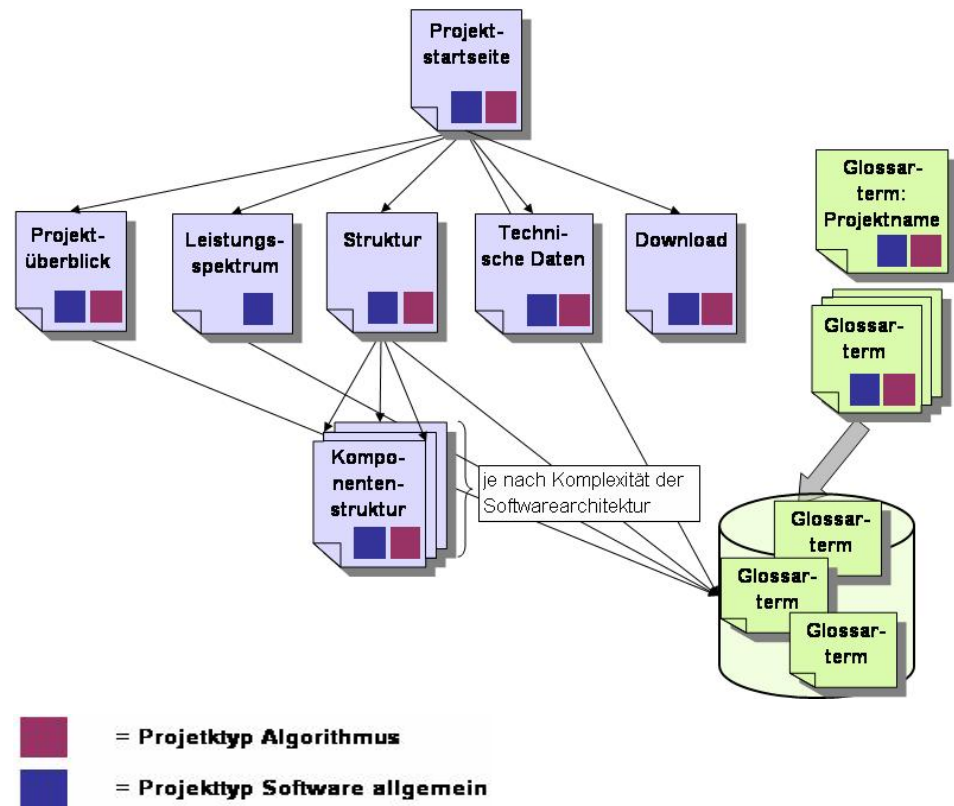


Abbildung 8-2: Struktur eines Projekt-Webauftrittes

Quelle: Eigene Darstellung

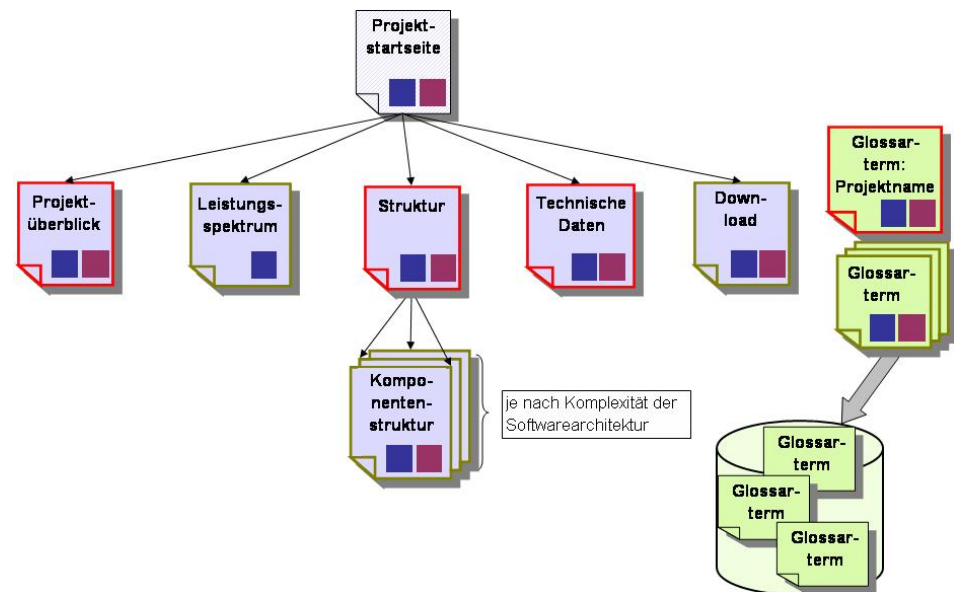
Zu Beginn ist ein Softwareprojekt nur durch die Kurzfassung der Projektübersichtsseite im Webauftritt vertreten. Sobald eine stabile Softwareversion vorliegt, dokumentiert der Autor sein Projekt im Idealfall mithilfe aller vorgesehenen Webseiten für seinen Projekttyp. Für *Softwareprojekte allgemein* sind dies:

- *Projektüberblick* in der Langfassung
- *Leistungsspektrum*
- *Struktur*
- *Technische Daten*
- *Download*
- Projektname als *Glossarterm*
- Eine einzelne Komponente der Softwarestruktur kann in Form einer neuen Webseite *Komponentenstruktur* detailliert beschrieben und über die Webseite *Struktur* verlinkt werden.

Projekte, die sich mit Algorithmen beschäftigen, unterscheiden sich zu den anderen Projekten in einer Webseite: *Leistungsspektrum* ist für sie nicht relevant, da sich ihre Funktionalität auf eine einzige beschränkt.

Obligatorische und optionale Webseiten

Um sicherzustellen, dass ein Projekt durch die wichtigsten Webseiten vorhanden ist und damit ein „rundes Bild“ von einem Projekt entsteht, sind obligatorische und optionale Webseiten für die beiden Projekttypen *Software allgemein* und *Algorithmus* definiert.



Legende

- = **Obligatorische-Projektwebseite**
- = **Optionale Projektwebseite**
- = **Projekttyp Algorithmus**
- = **Projekttyp Software allgemein**
- = **Optionale Glossarterm-Webseite**

Abbildung 8-3: Projekttyp abhängige optionale und obligatorische Webseiten

Quelle: Eigene Darstellung

Gründe für die Einstufung optional

Die Webseite *Download* ist als optional markiert, da Interessierte bei den Ansprechpartnern auf der Projektüberblicksseite nachfragen können. Auch besteht die Möglichkeit, dass z.B. Studenten ein Download ihrer Software nicht wünschen.

Die Webseite *Komponentenstruktur* ist als optional eingestuft, da sie in Abhängigkeit von der Komplexität der Software bzw. Softwarearchitektur erstellt wird. Diese Einschätzung unterliegt dem jeweiligen Autor.

Auf die Webseite *Leistungsspektrum* kann verzichtet werden, da die positiven Aspekte der Software auch auf der Webseite *Projektüberblick* zur Sprache kommen.

Auch Glossarterm-Webseiten zählen mit Ausnahme des Projektnamens zu den optionalen Webseiten. Diese werden angelegt, wenn Begrifflichkeiten verwendet werden, die je nach Vorwissen des Users einer Erklärung bedürfen, aber nicht direkt im Text erläutert werden sollen. Besonders Begriffe zum Thema *Grid-Computing* sind potentielle Glossartermkandidaten.

8.3.2.4 Webseitentypen

Jeder Webseite liegt ein *Webseitentyp* zugrunde, der im Feinkonzept beschrieben ist. Ein Webseitentyp übernimmt innerhalb des Webauftrittes eine bestimmte Funktion und ist für festgelegte Zielgruppen konzipiert:

Funktionen der
Webseitentypen

Webseitentyp	Funktion	Zielgruppen
Startseite	Überblicks-, Informationsseite, Marketing orientiert: - dient dem User als Orientierungs- und Navigationshilfe innerhalb des gesamten Webauftrittes der Grid-Softwaregruppe - weckt das Interesse des Users und erläutert kurz Grid-Computing	- Entwickler - Softwareuser - Administratoren - Investoren - Geschäftsführung - Interessierte
Ziele	Informationsseite, Marketing orientiert: - vermittelt dem User die Arbeitsziele der Grid-Softwaregruppe - soll beim User Interesse wecken für die Arbeit der Grid-Softwaregruppe	- Entwickler - Softwareuser - Administratoren - Investoren - Geschäftsführung - Interessierte
Aktuelle Projekte	Überblicks-/ Orientierungsseite, Marketing orientiert: - bietet dem User einen Überblick zu laufenden und abgeschlossenen Projekten des aktuellen und letzten Jahres - soll beim User Interesse wecken	- Entwickler - Softwareuser - Administratoren - Investoren - Geschäftsführung - Interessierte
Alte Projekte	Überblicks-/Orientierungsseite, Marketing orientiert: - bietet dem User einen Überblick zu abgeschlossenen Projekten, die älter als ein Jahr sind - soll beim User Interesse wecken	- Entwickler - Softwareuser - Administratoren - Investoren - Geschäftsführung - Interessierte
Aktuelle Publikationen	Überblicks-/ Orientierungsseite, Marketing orientiert: - bietet dem User einen Überblick zu Publikationen des aktuellen Jahres und der letzten beiden Jahre - enthält eine Suchfunktion	- Entwickler - Investoren - Geschäftsführung - Interessierte
Alte Publikationen	Überblicks-/Orientierungsseite, Marketing orientiert: - bietet dem User einen Überblick zu Publikationen, die drei Jahre und älter sind - enthält eine Suchfunktion	- Entwickler - Investoren - Geschäftsführung - Interessierte

Webseitentyp	Funktion	Zielgruppen
Glossar	Informationsseite, Nachschlagemöglichkeit: - dient dem User zum Nachschlagen von relevanten Begriffen rund um Grid-Computing - bietet ihm schnell und übersichtlich Zugriff auf verschiedene Terme	- Entwickler - Softwareuser - Administratoren - Interessierte
Kontakt	Informationsseite : bietet dem User kompakt verschiedene Möglichkeiten, die Grid-Softwaregruppe zu erreichen	- Entwickler - Softwareuser - Administratoren - Investoren - Interessierte
Projektstartseite	Überblicks-, Informationsseite, Marketing orientiert: - dient dem User als Orientierungs- und Navigationshilfe innerhalb eines Projektes - weckt das Interesse des Users und gibt ihm einen kurzen Einblick in das Projekt	- Entwickler - Softwareuser - Administratoren - Investoren - Geschäftsführung - Interessierte
Projektüberblick	Überblicks-, Informationsseite, Marketing orientiert: gibt dem User einen Überblick zum Projekt und verdeutlicht ihm seine Relevanz	- Entwickler - Softwareuser - Administratoren - Investoren - Geschäftsführung - Interessierte
Leistungsspektrum	Überblicks-, Informationsseite, Marketing orientiert: - gibt dem User einen Überblick zum Leistungsspektrum (Hauptfunktionalitäten) der Software - soll sein Interesse wecken, die Software auszuprobieren oder weitere Informationen anzufordern	- Entwickler - Softwareuser - Administratoren - Investoren - Interessierte
Struktur	Informationsseite, Überblicksseite: - gibt dem User einen Überblick zum Aufbau der Softwarearchitektur und erläutert sie - dient dem User als Entscheidungsgrundlage, ob er die Software testen will	- Entwickler - Softwareuser - Administratoren - Interessierte
Strukturkomponente	Informationsseite, Überblicksseite: - gibt dem User einen Überblick zum Aufbau einer Strukturkomponente und erläutert sie - dient dem User als Entscheidungsgrundlage, ob er die Software testen will	- Entwickler - Softwareuser - Administratoren - Interessierte
Technische Daten	Technischer Informationsüberblick/ Technisches Datenblatt: - zeigt dem User die Voraussetzungen auf, die seine Software und Hardware erfüllen muss, damit er die Projektsoftware nutzen kann - klärt die rechtlichen Fragen (Lizenzen)	- Entwickler - Softwareuser - Administratoren - Interessierte
Download	Shop/Informationsseite: - bietet dem User die Projektsoftware und die dazugehörige Dokumentation zum Download an, die vom Entwickler/Autor freigegeben sind - gibt zu jedem Download relevante Eckdaten	- Entwickler - Softwareuser - Administratoren - Interessierte

Webseitentyp	Funktion	Zielgruppen
Glossarterm	Informationsseite, Nachschlagemöglichkeit: ■ Bietet dem User „on the fly“ eine Begriffserklärung an	■ Entwickler ■ Softwareuser ■ Administratoren ■ Investoren ■ Geschäftsführung ■ Interessierte

Tabelle 8-3: Funktionen und Zielgruppen der Webseitentypen

Quelle: Eigene Darstellung

Strukturierungselemente für Webseiten

Der Webseitenaufbau zeichnet sich gemäß Usability-Empfehlungen durch einheitliche Strukturierungselemente aus:

- eine feste Navigation
- eine Überschrift mit Bild des Webseitentyps bei projektunabhängigen Webseiten, das mit den Bildern auf der Startseite identisch ist, um den User zu leiten. Die Webseiten eines Projektes enthalten stattdessen das Projektlogo, um alle zu einem Projekt gehörigen Webseiten auch optisch zu kennzeichnen. Ist keines vorhanden, rückt automatisch das Bild des Webseitentyps an seine Stelle.
- ein Slogan, der bei einem Projekt auf jeder seiner Webseiten erscheint und den Zweck des Projektes verdeutlicht
- eine kurze Einleitung, damit der User weiß, was ihn auf der Webseite erwartet
- Links, die für einen Webseitentyp festgelegt sind, damit der User gezielt navigieren kann (siehe Linkkonzept Kapitel 8.4)
- Unterüberschriften mit Link zum Seitenanfang, um diesen schnell zu erreichen
- Textblöcke mit max. 10 Zeilen und Leerräumen, um dem Auge des Users Ruhepausen zu gönnen
- Tabellen und Aufzählungslisten mit max. zwei Ebenen für eine übersichtliche Darstellung des Inhalts
- eine Fußzeile mit Link zum Seitenanfang sowie Eckdaten zur Webseite und rechtlichen Hinweisen

Layoutcharakteristika

Das Layout wird zentral geregelt, um Änderungen nur an einer Stelle durchführen zu müssen und bietet damit einen entscheidenden Vorteil zur Layoutumsetzung des FZK-Webauftrittes. Hier werden zahlreiche Stylesheets mit direkten Formatierungen im HTML-Code kombiniert und dadurch die Suche nach einer bestimmten Formatierung erschwert. Zudem wird die Copyright Identity des Forschungszentrums berücksichtigt, sei es z. B. hinsichtlich der Textgestaltung, der mittigen Platzierung oder der Satzspiegelbreite. Im Gegensatz zum FZK-Layout

wird auf die Strukturierung der Webseite mithilfe von Layouttabellen komplett verzichtet, um lange Ladezeiten zu umgehen (Schulz:2006).

Userfreundliche Bilder Die Tooltips für Bilder werden automatisch generiert und sind je nach funktionalem Zweck des Bildes individuell gestaltet. So erhält z. B. ein Bild, das eine Struktur erläutert, einen anderen Tooltip wie ein Projektlogo oder ein Bild, das einen Glossarbereich verdeutlicht. Für Bilder, die von der Grid-Softwaregruppe selbst erstellt werden, ist das komprimierte Bildformat *PNG* festgelegt, welches für das Internet geeignet ist und sich verlustfrei skalieren lässt.

Auch sind feste oder max. Werte für Größe und Breite der Bilder vorgegeben, um ein einheitliches Erscheinungsbild zu sichern. Für jedes Bild steht dem User ein Ersatztext zur Verfügung.

Datenschutz sichern Um zu verhindern, dass die Angabe einer E-Mail Adresse auf den Webseiten dazu führt, Spam-Nachrichten zu erhalten, wird zum einen die E-Mail-Adresse verschlüsselt übertragen, zum anderen das @-Zeichen und die Punktzeichen durch Bilder ersetzt. Es ist festgelegt, dass der Freigeber einer Webseite namentlich und mit seiner E-Mail Adresse in der Fußzeile optional aufgeführt wird. Aus Datenschutzgründen erhält er die Möglichkeit, auf jeder Webseite über beides selbst zu entscheiden, ebenso wie der Softwareentwickler, der nur namentlich genannt wird. Lizenzen und Copyright zur bereitgestellten Software und Dokumentation sowie den Bildern sind auf den Webseiten durch entsprechende Angaben vertreten. Die von der Autorin ausgesuchten Bilder, die die Webseitentypen symbolisieren, stammen von dem Bildanbieter *www.pixelio.de* und sind kosten- und lizenzfrei. Die Webseiten enthalten einen entsprechenden Hinweis auf den Anbieter.

Single Source Bei der Konzeption der Webseiten war der Single Source Gedanke hinsichtlich der Inhalte entscheidend, um den Schreibaufwand für die Grid-Softwaregruppe zu reduzieren, aber auch um die Konsistenz fester Textbausteine zu sichern. Es werden zwei verschiedene Vorgehensweisen für die Wiederverwendung von Inhalten verwendet:

Zum einen werden Inhalte vom User einmal angegeben und auf anderen Webseiten wieder verwendet. Eine zentrale Rolle spielt dabei die Übersichtsseite, durch die jedes Projekt von Beginn an vertreten ist. Sie wird deshalb dazu genutzt, Angaben vom User anderen Webseiten zur Verfügung zu stellen:

Beispiele für Single Source Dazu zählt z.B. der Slogan zu einem Projekt, der auf allen Projektwebseiten sowie den beiden automatisch generierten Webseiten *Aktuelle* und *Alte Projekte* aufgeführt wird. Auch die Zustimmung eines Softwareentwicklers, ob er namentlich genannt werden will, spielt für andere Webseiten eine Rolle. In Abhängigkeit davon erscheint sein Name im Inhalt des Metadatums *Keywords* auf jeder seiner

Projektwebseiten sowie im Content der beiden Webseiten *Aktuelle* und *Alte Projekte*, der Projektdownloadseite und der Strukturwebseite eines anderen Projektes. Gibt der Autor von der Strukturwebseite das Projekt an und vermerkt es als Entwicklung der Grid-Softwaregruppe, wird automatisch überprüft, ob der Entwickler auf seiner Übersichtsseite der Namensveröffentlichung zugestimmt hat und wird entsprechend seiner Angabe aufgeführt oder nicht. Eine umständliche Suche nach dem Projektordner und der entsprechenden Datei entfällt. Wie der Softwareentwickler sind auch die auf einer Projektübersichtsseite angegebenen Kontaktpersonen ein Teil des Metadatums *Keywords* und damit in jeder dazugehörigen Projektwebseite enthalten. Auch die Kurzbeschreibung der Übersichtsseite wird optimal wieder verwendet. Sie wird in einem Teaser der Projektstartseite zur Vorstellung des Projektes genutzt, damit der User schnell entscheiden kann, ob ihn das Projekt interessiert. Gleichzeitig wird der Text als Definition für den automatisch generierten Glossarterm zum Projektnamen eingesetzt. Neben Texten werden auch Bilder wieder verwendet: Das Projektlogo wird ebenfalls auf allen anderen Projektwebseiten angezeigt.

Automatisch eingefügte Textmodule

Zum anderen beinhalten alle Webseiten feste Textmodule: Dazu zählen Überschriften, Einleitungen, Linktexte, Tooltips bei Links und Bildern, Copyrighttexte, Schlüsselwörter, Metadaten, Fußzeilentexte sowie Tabellenzeilen- bzw. Tabellenspaltenbeschriftungen. Die Textmodule werden automatisch bei der Transformation der XML-Webseite an den entsprechenden Stellen eingefügt.

8.3.2.5 Metadaten der Webseitentypen

„Seiten ohne Metadaten ins Web zu stellen ist wie wenn man seine Werbebroschüren im eigenen Hinterhof ablegt: Sie werden allenfalls durch Zufall gefunden.“ (Jakobsen 2003)

Fenstertitel

Um sicherzustellen, dass die Webseiten im WWW durch Suchmaschinen gefunden werden, wird als Titelzeile des Browserfensters die Webseitenüberschrift verwendet.

Metaangaben

Zudem kombiniert die Autorin normale META-Angaben mit einigen Dublin Core Angaben. Laut Lubkowitz beeinträchtigen sich diese Angaben gegenseitig nicht (2005:255). Gerade Schlüsselwörtern kommt für das Auffinden von Webseiten im WWW eine elementare Rolle zu. Um eine Mindestzahl an Schlüsselwörtern zu gewährleisten, werden neben dem automatisch eingefügten Namen des Softwareentwicklers sowie den Kontaktpersonen zu einem Projekt auch Webseiten übergreifende Keywords automatisch eingefügt, die als Textmodul ausgelagert sind. Gleichzeitig wurde festgelegt, dass pro Webseite jeweils mindestens ein Schlüsselwort aus den beiden Bereichen Technologie und wissenschaftliche Leistung vorhanden sein muss – zwei wichtige Aspekte in der Forschung.

8.4 Linkkonzept

8.4.1 Vorgehensweise bei der Konzeption

*„Klick mich!“ – „Nein mich!“ „Wenn du mich nicht klickst verpasst du was“
(Achtelig 2005:45)*

Um Konsistenz im Setzen von Verlinkungen zu sichern, wurde parallel zu Erstellung der Webseitentypen ein Konzept für Linkarten entwickelt und in Tabellenform festgehalten (siehe CD Datei *Webseiten_und_Linktypen.xls*). Jeder Link wird dabei hinsichtlich bestimmter Merkmale beschrieben, die standardisiert sind.

Einteilungskriterien Linkmatrix

Diese Kriterien umfassen Daten wie ...

- Linktypname
- Linktyp-ID mit ID-Schema
- Quell- und Ziel-Webseitentyp
- Definition des Linktyps
- Art der Linkverbindung (Intra-, Inter-, Extra-Webseitenlink)
- Vorkommen (optional, erforderlich)
- Darstellung (z. B. Linktext, Linksymbol, Tooltip, Beispiel) und Position des Linktyps innerhalb einer Webseite
- technische Umsetzung

Um auch bei der Festlegung der Linktypen und ihrer Benennung standardisiert vorzugehen, wurden Namensregeln, Einteilungskriterien für Linktypen sowie Bedingungen festgelegt, bei deren Erfüllung ein neuer Linktyp angelegt werden darf.

8.4.2 Ergebnis des Linkkonzeptes

Hauptnavigation

Zur Navigation steht dem User stets das Navigationsmenü als fester Bestandteil aller Webseiten zur Verfügung, außer bei Glossarterm-Webseiten. Die statische Navigationsleiste bietet jedoch nur Zugriff auf die Webseiten der obersten Ebene im Webauftritt.

Teaser

Deshalb kommt den Teasern, aus denen die Start- und Projektstartseite bestehen, eine wichtige Bedeutung zu. Jede Webseite, die sich direkt unter einer Startseite befindet, ist durch einen Teaser vertreten und mit diesem verlinkt, so dass der User trotz fehlender Navigationsmöglichkeit über die Navigationsleiste auch in tiefere Ebenen navigieren kann.

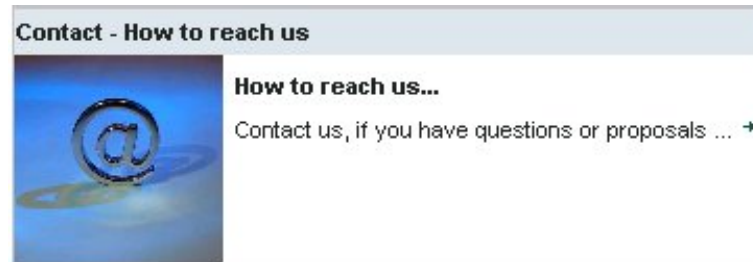


Abbildung 8-4: Teaser der Startseite auf die Kontaktseite

Quelle: Eigene Darstellung

Zudem lässt sich jeder Teaser ein- und ausschalten und damit eine Webseite zu- und abschalten. Gerade bei Projektauftritten, bei denen Webseiten fehlen können, erhält der User trotzdem ein kompaktes Bild.

Konsistentes Erscheinungsbild

Jeder Link ist eindeutig und damit userfreundlich gestaltet. Dies erreicht die Kombination aus aussagekräftigem Linktext oder Linkbild und Tooltip, die den User über den Zweck bzw. das Ziel des Links informieren. Ergänzt wird dies zum Teil durch eine einführende Linkinformation. Das konsistente Erscheinungsbild aller Links eines Linktyps sorgt im Webauftritt für Klarheit. Der User erkennt auf diese Weise schnell, ob sich das Linkziel in einem neuen Fenster öffnet oder im selben. Besuchte Links ändern zudem ihre Farbe und geben dem User einen Anhaltspunkt, welche Inhalte er bereits gelesen hat.

Linklisten statt verlinkter Fließtexte

Um den User innerhalb eines Fließtextes nicht durch Links unnötig abzulenken, sind nur Glossarbegriffe verlinkt, die sich in einem Pop-up Fenster öffnen. Anstatt auf das Glossar wird auf eine eigene Webseite des Glossarbegriffes verlinkt, um dem User nur einen Informationsausschnitt anzubieten. Weiterführende Informationen zu einem Thema werden immer über eine Linkliste am Seitenende bzw. am Ende eines Abschnittes angeboten.

Linktypen und ihre Funktion im Webauftritt

Im Folgenden sind die einzelnen Linktypen kurz hinsichtlich ihrer Funktion im Webauftritt erläutert.

Linktyp-ID	Funktion/Ziel	Verbindung
top_link	Link zum Seitenanfang, der Unterabschnitte und Fußzeile innerhalb einer Webseite mit dem Seitenanfang verbindet und damit dem User ein schnelles Navigieren innerhalb einer Webseite ermöglicht.	Intra-Webseiten Link
section_link	Link einer Linkliste, der auf einen Seitenabschnitt bzw. Unterüberschrift innerhalb der Webseite verlinkt. Die Linkliste wird ab zwei Unterüberschriften automatisch generiert, damit der User gezielt auf einzelne Bereiche einer längeren Webseite zugreifen kann.	Intra-Webseiten Link
preferred_term_section_link	Link, der von einem Synonym innerhalb des Glossars auf dessen Vorzugsbenennung verlinkt. Der User erhält so schnell die korrekte Benennung und Erklärung des Begriffes.	Intra-Webseiten Link

Linktyp-ID	Funktion/Ziel	Verbindung
index_link	Link, der von den Webseiten <i>Aktuelle Projekte</i> und <i>Alte Projekte</i> auf eine Projektstartseite verlinkt. Der User erhält die Möglichkeit, auf die einzelnen Projekte zuzugreifen.	Inter-Webseiten Link
term_index_link	Link, der vom Glossareintrag zum Projektnamen auf die entsprechende Projektstartseite verlinkt. Der User erhält so detaillierte Informationen.	Inter-Webseiten Link
topic_link	Link, der von einer Startseite aus auf eine Projektseite oder eine projektunabhängige Webseite (<i>Thema</i>) eine Ebene tiefer verlinkt. Der User kann auf die einzelnen Inhalte zum Projekt zugreifen.	Inter-Webseiten Link
old_topic_link	Alleinstehender Link am Seitenende einer aktuellen Webseite, der auf eine Webseite mit alten Themen verlinkt. Der User kann im „Archiv“ weitersuchen.	Inter-Webseiten Link
sub_topic_link	Link einer Linkliste, der auf eine Webseite eine Ebene tiefer verlinkt. Er befindet sich am Ende eines Unterabschnittes bzw. der Webseite und führt den User zu detaillierten Informationen des behandelten Themas.	Inter-Webseiten Link
menue_topic_link	Link im Navigationsmenü, der auf die obersten Webseiten des Auftritts verlinkt. Der User kann von jeder Webseite aus schnell auf die obersten Webseiten zugreifen.	Inter-Webseiten Link
glossaryterm_topic_link	Link, der direkt im Text die Erklärung eines Fachbegriffes anzeigt. Der User kann nach Bedarf sich die Begriffserklärung einblenden lassen.	Inter-Webseiten Link
www_link	Link, der allein oder als Teil einer Linkliste auf eine externe Webseite im WWW verlinkt. Der User erhält Zusatzinformationen zu einem Thema.	Extra-Webseiten Link
member_www_link	Alleinstehender Link, der auf die Homepage eines externen Projektmitgliedes verlinkt. Der User bekommt Informationen zu den anderen Projektbeteiligten.	Extra-Webseiten Link
footer_member_www_link	Alleinstehender Link in der Fußzeile, der auf die Homepage eines externen Dienstleisters verlinkt, um die rechtlichen Bedingungen zu erfüllen.	Extra-Webseiten Link
download_link	Link, über den der User etwas herunterladen kann, z. B. eine Lizenz. Er steht allein oder als Teil einer Linkliste.	Extra-Webseiten Link
explained_download_link	Alleinstehender Link, über den der User etwas herunterladen kann, der aber Zusatzinformationen enthält wie Lizenzangaben (z. B. beim Download von Software)	Extra-Webseiten Link
publication_download_link	Alleinstehender Link, über den der User eine Publikation herunterladen kann. Diese Seiten werden aus LaTeX heraus generiert und sind deshalb im Linkkonzept als eigener Link vertreten.	Extra-Webseiten Link
email_link	Alleinstehender Link, über den sich eine E-Mail Adresse im Outlook-Fenster öffnet. Der User kann schnell und unkompliziert eine E-Mail an Ansprechpartner senden.	Extra-Webseiten Link

Tabelle 8-4: Linktypen und ihre Funktion im Webauftritt

Quelle: Eigene Darstellung

Beziehungen der Linktypen

Die Beziehungen der Linktypen zueinander sind grafisch dargestellt (siehe Anhang H).

9 Technische Realisierung des Webauftrittskonzeptes

9.1 Einführung

Technische
Komponenten für
einen Webseitentyp

Jeder Webseitentyp wird technisch umgesetzt mithilfe ...

- eines Schemas, das modulare Schemas referenziert
- einer XML-Templatedatei
- eines XSLT-Stylesheets, das modulare Stylesheets referenziert
- von Textmodul-Einträgen in der XML-Datei für Textmodule
- von Stilen in einer CSS-Datei
- eines SPS-Stylesheets zur Anzeige der Webseite im Editor *Authentic*

Zudem steht für jedes Softwareprojekt sowie für alle projektunabhängigen Webseiten gemeinsam je eine SPP-Projektdatei zur Verfügung.

Alle Webseiten, die die Navigationsleiste enthalten, werden mit XSLT in PHP transformiert. Einzig die Webseiten für Glossarterme ohne Navigation haben das Outputformat HTML.

Technische
Realisierung auf CD

Im Folgenden kann nur ein Auszug der Realisierungen dargestellt werden. Die gesamte technische Realisierung mit allen Dateien befindet sich auf beiliegender CD im Ordner *GRID_projects*.

9.2 Realisierung der Inhaltsstruktur mit eigenem Informationsmodell

Für eine konsistente inhaltliche Strukturierung der Webseiten und Sequenzierung der Informationen wurde ein eigenes Informationsmodell konzipiert und als Schema in XML umgesetzt. Zwar ist das standardisierte Informationsmodell *DocBook* für die Dokumentation von Software geeignet wie auch das speziell für die Publikationen in Online Medien eingesetzte *DITA* (siehe Kapitel 3.9), beide werden aber primär aufgrund der folgenden Gründe nicht eingesetzt:

Gründe für ein eigenes
Informationsmodell

DocBook ist zu umfangreich und für einen schnellen Einsatz ungeeignet: “[I]f you’re looking for a simple model, DocBook probably won’t be your solution. It is a very complex model and although you can simplify it for your users, it takes considerable effort to learn and understand.” (Rockley 2003:178). *DITA* befindet sich hinsichtlich Komplexität und Aufwand im mittleren Bereich (vgl. Ziegler 2006:103) und weist für marketingorientierte Webseiten kein Repertoire an Elementen auf.

Strukturierungs-
methode: primär
Funktionsdesign

Als Strukturierungsmethode wird eine Kombination aus Funktionsdesign und layoutorientiertem Ansatz eingesetzt. Die Elemente sind primär nach ihrer kom-

munikativen Funktion benannt. Eine Unterscheidung nach der Layoutfunktion erfolgt lediglich auf den untersten Ebenen.

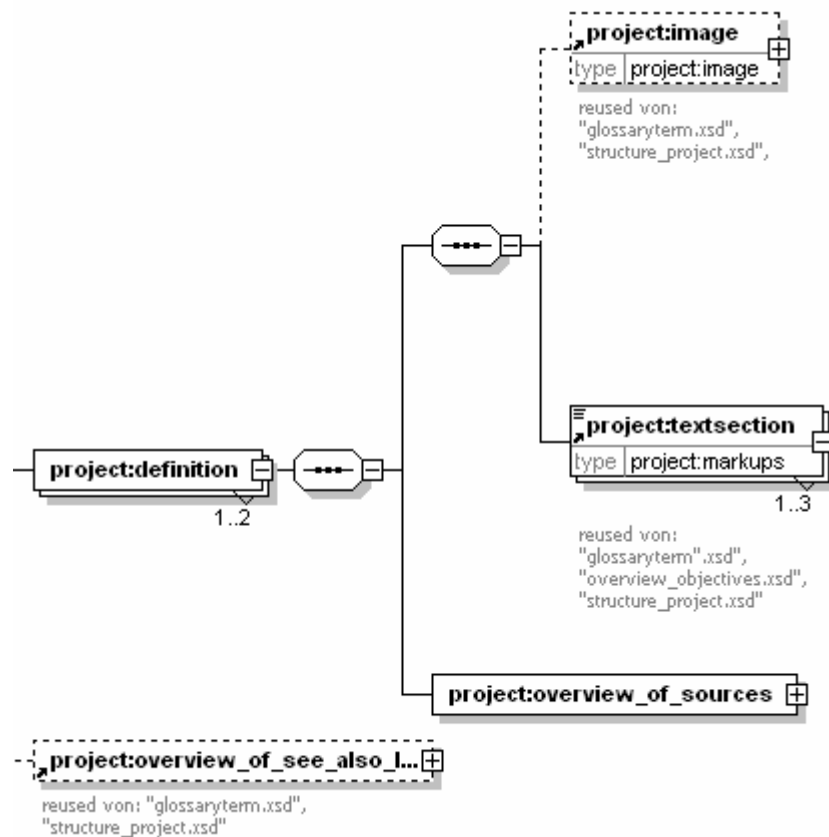


Abbildung 9-1: Elementbenennung nach Funktionsdesign und Layout

Quelle: Codeausschnitt aus Schema *glossaryterm.xsd*

Die Elemente identifizieren sich direkt über ihren Namen oder mit Hilfe ihrer Positionierung im *Ancestor*-Pfad selbst, damit für den Autor ohne lange Einarbeitung ersichtlich ist, welche Informationen von ihm gefordert werden. Bei Elementen mit mehreren Spezifizierungsmöglichkeiten ist nicht für jede ein eigenes Element angelegt, sondern das Element wird über die Werte eines Attributes charakterisiert, wie z. B. bei den Elementen *see_also_link* oder *markup*. Die Elemente und Attribute sowie *Complex Types* sind konsistent benannt und weisen englische Namen auf.

Namensraum

Die Eindeutigkeit der Element- und Attributnamen in den XML-Dokumenten, die zum Webaufttritt der Grid-Softwaregruppe gehören, sichert ein Namensraum. Die Zuordnung zum Namensraum erfolgt über das Präfix *project*.

Patterns und Enumerations

Wenn möglich werden *Enumerations* zur Datentypeinschränkung verwendet, um dem User die Dateneingabe zu erleichtern. Für Usereingaben, die fehleranfällig und gleichzeitig einem bestimmten Schema unterliegen, ist der *Simple Type* des Elements mithilfe von Patterns eingeschränkt wie bei den beiden Elementen *email* und *phone*.

project:email		project:phone	
type	xs:string	type	xs:string
derivedBy	restriction	derivedBy	restriction
pattern	(.*)@(.*)\.de	pattern	^+49\s{0}[1-9]{4}\s[0-9]{2...

Abbildung 9-2: Elemente *email* und *phone* mit eingeschränktem Datentyp *string*Quelle: Codeausschnitt aus Schema *contact.xsd*

Modularisierung und Referenzierung von Schemas

Jeder Webseitentyp wird durch ein eigenes Schema vertreten, um möglichst kleine Schemas für eine bessere Übersichtlichkeit zu erhalten. Globale Attribute, Elemente und *Complex Types*, die in zwei oder mehreren Schemas vorkommen, sind in separate Schemas ausgelagert. Typische „Kandidaten“ für die Auslagerung sind alle Elemente, denen eine Metadatenfunktion in den XML-Dokumenten zukommt. Sie sind in dem Schema *share_metadata.xsd* abgelegt. Alle anderen gemeinsamen Attribute, Elemente und Strukturen finden sich in *share.xsd*. Während *share_metadata.xsd* von *share.xsd* referenziert wird, wird dieses wiederum von jedem Schema der Webseitentypen referenziert. Auf diese Weise stehen einem Webseitentyp-Schema alle ausgelagerten Elemente und *Complex Types* zur Verfügung.

Complex Types

Complex Types dienen dazu, Informationsgebilde wieder zu verwenden und sind folglich für die Schemamodularisierung elementar. Wenn sinnvoll sind bei ähnlichen Schemastrukturen *Complex Types* durch Einschränkungen oder Erweiterungen voneinander abgeleitet – eine Vorgehensweise, die die *Complex Types* der Metadaten demonstrieren (siehe Anhang I). Da das Schema im Gegensatz zur DTD keinen Datentyp für leere Elemente vorsieht, wurde der *Complex Type empty* angelegt und als neuer Datentyp definiert. Dieser enthält keine Kindelemente und entspricht damit dem Datentyp EMPTY in der DTD. Von *empty* sind die drei *Complex Types image*, *file* und *link* abgeleitet, indem Attribute- bzw. Attributgruppen ergänzt wurden. Jeder *Complex Type* wird als Datentyp für mehrere Elemente verwendet, wie z. B. *link* als Datentyp für alle Elemente, die einen Link repräsentieren.

9.3 Realisierung von XML-Templates

Auf Grundlage des jeweiligen Schemas ist jeder Webseitentyp durch ein XML-Template vertreten, wobei sich entsprechend des Automatisierungsgrades der Webseiten zwei Arten unterscheiden lassen:

- Templates für Webseiten, die vom Autor manuell mit Inhalt befüllt werden
- Templates von automatisch erstellten Webseiten, deren Content vollständig über die in Kapitel 6.3 beschriebenen Automatisierungsprozesse eingetragen wird

Templates für Webseiten, die vom Autor manuell mit Inhalt befüllt werden, beinhalten Beispieltexte und Anweisungen, um den Autor zu führen und ihm das Schreiben zu erleichtern. Sollte später ein anderer Editor anstelle des *Authentic* eingesetzt werden und damit die Hilfestellungen entfallen, die mit *Authentic* möglich sind (siehe Kapitel 9.8), erhält der Autor dennoch Anhaltspunkte für seinen Text. Zusätzlich enthalten sie analog zur anderen Templateart Platzhalter, an deren Stelle über Automatisierungsprozesse Inhalte eingefügt werden.

9.4 Modularisierung der XSLT-Stylesheets und Beispiele

Das XSLT-Stylesheet jedes Webseitentyps referenziert modulare Stylesheets, die wieder verwendbaren Code enthalten:

Referenzierung modularer Stylesheets

```

<!-- Einbinden von Stylesheets, die reusable Templates/Funktionen enthalten*****
<!-- Einbinden des Templates: contacts_generate.xml*****
<xsl:include href="/contacts_generate.xml"/>
<!-- Einbinden des Templates für Fusszeile*****
<xsl:include href="/footer_generate.xml"/>
<!-- Einbinden des Templates für Grafiken*****
<xsl:include href="/graphic_generate.xml"/>
<!-- Einbinden des Templates für Headlines und Subheadlines*****
<xsl:include href="/headlines_generate.xml"/>
<!-- Einbinden des Templates für Metadatenabschnittes*****
<xsl:include href="metadata_generate.xml"/>
<!-- Einbinden des Templates für Navigation*****
<xsl:include href="/navigation_insert.xml"/>
<!-- Einbinden des Templates für Variablen, die allgemeingültige Textmodule enthalten**
<xsl:include href="/shared_textmodule_variables.xml"/>
<!-- Einbinden des Templates für Tabelle *****
<xsl:include href="/table_generate.xml"/>
<!-- Einbinden der Templates: text_specials.xml (Auszeichnungselemente)*****
<xsl:include href="/text_markup.xml"/>
<!-- Linkarten*****
<!-- Einbinden des Linktyps: glossaryterm_topic_link*****
<xsl:include href="/glossaryterm_topic_link.xml"/>
<!-- Einbinden des Linktyps: email_link*****
<xsl:include href="/email_link.xml"/>
<!-- Einbinden des Linktyps: section_link (Seitenabschnitt)*****
<xsl:include href="/section_link.xml"/>
<!-- Einbinden des Linktyps: top_link (Seitenanfang)*****
<xsl:include href="/top_link.xml"/>

```

Abbildung 9-3: Referenzierung der modularen Stylesheets im Stylesheet overview_project.xml

Quelle: Codeausschnitt aus XSLT-Stylesheet overview_project.xml

„Kandidaten“ für modulare Stylesheets

Zu den Komponenten, die in solche separaten Stylesheets ausgelagert sind, zählen typische Strukturierungs- bzw. Layout- sowie Content-Elemente:

- Über- und Unterüberschrift (*headlines_generate.xml*)
- Fußzeile (*footer_generate.xml*)
- Navigation (*navigation_insert.xml*)

- Tabellen (*table_generate.xml*)
- Bilder mit Tooltips (*graphic_generate.xml*)
- Linkarten wie der Link auf eine Glossartermdatei (*glossary-term_topic_link.xml*), einen Seitenabschnitt (*section_link.xml*), zum Seitenanfang (*top_link.xml*) oder der E-Mail Link (*email_link.xml*)
- Textauszeichnungen (*text_markup.xml*)
- Metadaten (*metadata_generate.xml*)
- Kontaktabschnitte (*contacts_generate.xml*)

Des Weiteren befinden sich globale Variablen für Inhalte, die aus einer Textmoduldatei ausgelesen werden, in einem separaten Stylesheet (*shared_textmodules_variables.xml*).

Vorteile von modularen Stylesheets

Entsprechend des Single Source Gedanken wird mithilfe solcher modularen Stylesheets eine doppelte Codehaltung vermieden. Zudem müssen Änderungen nur an einer Stelle durchgeführt werden, was die Pflege der Stylesheets erleichtert. Auch wird ein einheitliches Erscheinungsbild bestimmter Strukturierungselemente durch die Verwendung desselben Codeabschnittes auf allen Webseiten gesichert. Bei der Umsetzung der zahlreichen Webseiten in Form von XSLT-Stylesheets zeigte sich, dass sich neue Stylesheets auf diese Weise schnell und effizient aus den vorhandenen modularen Stylesheets zusammenbauen lassen und der Aufwand reduziert wird.

Voraussetzung für den Einsatz solcher Stylesheetmodule ist eine einheitliche Strukturierung der Webseiten, die zum Teil durch Schemavorgaben mithilfe von globalen Elementen und *Complex Types* umgesetzt werden (siehe Kapitel 9.2).

Die modularen Stylesheets sind analog zur Realisierung der Webseitentypen in Tabellenform beschrieben (siehe Anhang J)

Zusätzlich werden noch andere Methoden für die Wiederverwendung von XSLT-Code eingesetzt:

Globale Variablen

Als eine weitere Methode, um die Stylesheets modular aufzubauen, werden globale Variablen verwendet, die in einem markierten Bereich im Kopf des Stylesheets gesammelt sind:

Um zu verhindern, dass Veränderungen in der Ordnerstruktur oder die Verschiebung von Dateien zu aufwändigen Änderungsarbeiten in einer XSLT-Datei führen, sind die Dateipfade zu den für die Webseiten wichtigen Ordnern *share*, *glossary* und *websites* als globale Variablen festgehalten (siehe Abbildung 9-4). In

den globalen Variablen werden außerdem die in Abbildung 9-4 dargestellten Dateipfade von Bildern und CSS- sowie JavaScript-Dateien gespeichert.

```

<!--Variable für Dateipfadteil zum Ordner share und glossary-->
<xsl:variable name="file_path_part">
  <xsl:choose>
    <xsl:when test="$website_id='index'">
      <xsl:value-of select=".."../>
    </xsl:when>
    <xsl:otherwise>
      <xsl:value-of select=""/>
    </xsl:otherwise>
  </xsl:choose>
</xsl:variable>

<!--Dateipfad zur externen CSS- und JavaScript-Datei*****-->
<xsl:variable name="css_path" select="
concat(normalize-space($file_path_part),'share/stylesheets/grid_web.css')"/>
<xsl:variable name="glossaryterm_js_path" select="
concat(normalize-space($file_path_part),'share/scripts/generate_glossaryterm_wi
ndow.js')"/>

<!--Variablen für verwendete Photos/Symbole*****-->
<xsl:variable name="photo_grid_teaser" select="
concat(normalize-space($file_path_part),'share/images/photos/grid_gridgroup_net
.jpg')"/>
<xsl:variable name="symbol_topic_link" select="
concat(normalize-space($file_path_part),'share/images/symbols/topic_link_arrow.
gif')"/>
<!--*****Ende des Stylesheet-Kopfes*****-->

```

Abbildung 9-4: Codeabschnitt der globalen Variablen im Stylesheet *index_index_project.xsl*

Quelle: Codeausschnitt aus XSLT-Stylesheet *index_index_project.xsl*

Ebenso werden die Variablen für Dateipfade von XML-Webseitendateien benutzt, aus denen Inhalt referenziert wird, oder für Textmodule, die aus einer Textbausteindatei ausgelesen werden. Daneben sind XPath-Ausdrücke und Werte als globale Variablen abgelegt, die innerhalb des Stylesheets mehrmals wiederholt werden.

Lokale Variablen

Selbst auf Ebene der Vorlagen und benannten Vorlagen wird der Gedanke der Modularisierung fortgesetzt. Es werden lokale Variablen eingesetzt, um für Werte, die in diesen Abschnitten öfters vorkommen, einen wieder verwendbaren „Behälter“ zu bilden.

Separate Textmoduldatei

Die in Kapitel 8.3.2.4 (S. 64) aufgeführten Textmodularten wie Tabellenzeilenbeschriftungen sind zentral in einer separaten XML-Textmoduldatei in standardisierter Form abgelegt und werden bei der Transformation der XML-Webseite an die entsprechenden Stellen referenziert. Dadurch können Änderungen zentral

durchgeführt werden und der Inhalt der XSLT-Dateien wird nicht unnötig „aufgebläht“.

Eine weitere Möglichkeit Textbausteine außerhalb der XSLT-Stylesheets anzulegen bieten separate XSLT-Dateien für die einzelnen Webseitentypen, die globale Variablen mit Textbausteinen als Werte enthalten. Für webseitenübergreifende Textmodule kann ein weiteres XSLT-Stylesheet bereitgestellt werden. Diese Variante weist jedoch zur XML-Textmoduldatei entscheidende Nachteile auf und wurde deshalb nicht angewandt:

- Die Textbausteine sind auf mehrere XSLT-Dateien verteilt, so dass bei Änderungen sowohl das Webseitenstylesheet als auch das webseitenübergreifende Stylesheet überprüft werden müssen.
- Um ein Textmodul einer globalen Variablen als Wert zuzuweisen, wird die Abfrage der Sprache und anderer Werte benötigt. Dies führt zu unübersichtlichen Codeabschnitten.
- Den Variablen in den Stylesheets können im Gegensatz zur Textmoduldatei die Werte über den Editor *Authentic* nur durch aufwändige Programmierung zugewiesen werden.
- Sollen Textbausteine innerhalb von Schleifen bei der XSLT-Transformation ausgelesen werden, muss bei jedem Schleifendurchgang überprüft werden, welcher Textbaustein eingebaut werden soll – eine umständliche Methode.

Beispiel: Separate Navigation einbinden

Für das Einbinden der Navigationsleiste wird in der separaten XSLT-Datei ein benanntes Template verwendet, das durch Aufruf in der XSLT-Webseitendatei, ausgeführt wird. Die Navigationsleiste ist als separate HTML-Datei umgesetzt, die mithilfe von PHP in die Webseiten eingebunden wird. Bei Änderungen an den Verlinkungen der Navigation muss lediglich die HTML-Datei angepasst werden. Die einzelnen Webseiten hingegen benötigen keine neue Generierung, um die Änderungen zu übernehmen.

```

<!--Navigation als fertige HTML-Datei über PHP einbinden*****-->
<xsl:template name="navigation_insert">
  <xsl:variable name="navigation_file" select="
concat($file_path_part,'share/navigation/navigation_', $language, '.html')"/>
  <script language="php">
    <xsl:text>require("</xsl:text>
    <xsl:value-of select="$navigation_file"/>
    <xsl:text>");</xsl:text>
  </script>
</xsl:template>

```

Abbildung 9-5: Einbindung der Navigation über PHP

Quelle: Codeausschnitt aus XSLT-Stylesheet *navigation_insert.xsl*

9.5 Realisierung von automatisierten Webseiten

9.5.1 Automatisch eingefügte Inhalte

Automatisierung durch
Textmoduldatei

Die zentrale Textmoduldatei dient nicht nur zur Modularisierung der XSLT-Stylesheets, sondern bildet die Quelle für das automatische Einfügen fester Textbausteine in die Output-Datei. Mehr noch: Anstatt den Text statisch innerhalb eines Templates in der XSLT-Datei einzufügen, gelingt durch konsistenten Aufbau und Benennungssystematik der XML-Textmoduldatei ein dynamisches Referenzieren der Inhalte innerhalb von Schleifen. Auch bei Templates, die für mehrere Elemente verwendet werden, reicht dieselbe Codezeile aus, um für die verschiedenen Elemente die entsprechenden Inhalte aus der Textmoduldatei auszulesen. Vor allem das Einfügen von Tabellenzeilen- oder Tabellenspaltenbeschriftungen wird durch dieses Verfahren vereinfacht.

Abbildung 9-6 verdeutlicht den Aufbau der Textmoduldatei.

Aufbau der Textmoduldatei

```

<!-- Ziele: objectives -->
<project:type_of_website project:name="objectives" -->
  <project:type_of_textmodules project:name="headline" -->
    <project:specification_of_type project:name="website" -->
      <project:textmodule project:language="de">Ziele der Grid-Softwaregruppe</project:textmodule>
      <project:textmodule project:language="en">Objectives of the Grid computer scientists
    </project:textmodule>
    </project:specification_of_type>
  </project:type_of_textmodules>

  <project:type_of_textmodules project:name="subheadline" -->
    <project:specification_of_type project:name="list_of_objectives" -->
      <project:textmodule project:language="de">Unsere Ziele sind...</project:textmodule>
      <project:textmodule project:language="en">Our objectives are...</project:textmodule>
    </project:specification_of_type>
  </project:type_of_textmodules>
</project:type_of_website>

```

Abbildung 9-6: Eintrag für Webseitentyp *Ziele* in der Textmoduldatei

Quelle: Codeausschnitt aus XML-Datei *textmodules.xml*

Jeder Webseitentyp (a) ist mit einem eigenen Abschnitt in der Textmoduldatei vertreten. Ein Abschnitt ist in Unterabschnitte gegliedert, wobei jeder Unterabschnitt für einen Textmodultyp steht. Dazu zählen z. B. Überschriften und Unterüberschriften (b). Ein Textmodultyp enthält weitere Abschnitte (c, d) mit Textmoduleinträgen in deutscher und englischer Sprache.

Textmoduleinträge am Beispiel des Webseitentyps *Ziele*

Abbildung 9-6 erläutert die Umsetzung von Textmodulen anhand des Webseitentyps *Ziele*. Der Webseitentyp wird über seine ID *objectives* als Attributwert des Elementes *type_of_website* angegeben. Die beiden Textmodultypen *Überschrift* und *Unterüberschrift* sind jeweils durch das Element *type_of_textmodules* vertreten. Analog zum Webseitentyp wird der Wert des Attributes *name* dazu verwendet, den Textmodultyp zu definieren. Die beiden Attributwerte *headline* und *subheadline* repräsentieren den jeweiligen Textmodultyp *Überschrift* und *Unterüberschrift*. Damit keine fehlerhaften Werte für die Webseiten- und Textmodultypen erfasst werden, sind die Eingabemöglichkeiten jeweils als Enumeration hinterlegt. Da die Unterüberschrift innerhalb der Webseite an mehreren Stellen vorkommen kann, muss die Stelle genau festgelegt werden, an der der Textbaustein platziert werden soll. Als Markierung dient das Webseitenelement *list_of_objectives*. Sobald das XSLT-Stylesheet des Webseitentyps bei der Transformation das Element *list_of_objectives* erreicht, wird der Textbaustein eingefügt. Findet sich im Schema des Webseitentyps kein entsprechendes XML-Element, das den Textbaustein in der Dokumentenstruktur vertritt, muss wie im Falle der Überschrift ein passender Wert gefunden werden. Der Attributwert *website* drückt aus, dass es sich um die Überschrift der Webseite handelt. Bei der XSLT-Transformation wird das Textmodul an der Stelle eingefügt, wo es der Informationsdesigner im Stylesheet angibt.

Neben den Abschnitten zu den einzelnen Webseitentypen gibt es einen Bereich *shared*, der Textmodule umfasst, die für mehrere Webseiten gelten. Dazu zählen unter anderem Schlüsselwörter für die Metadaten, Tooltips für Links oder Texte für die Fußzeile.

Automatisches
Befüllen von
Tabellenzellen

Das dynamische Einfügen der Textmodule beim Aufbau von Tabellen über Schleifen wird z. B. beim Stylesheet für den Webseitentyp *Technische Daten* mehrmals eingesetzt:

```
<!--Abschnitt für Software-Voraussetzungen mit
Tabelle----->
<xsl:template match="software_requirements">
  <div class="website_position">
    <xsl:call-template name="subheadline_generate"/>
    <!--Tabelle mit automatisch eingefügter
Tabellenzeilenbeschriftung----->
-->
    <table class="website_table_general">
      <xsl:for-each select="/*">
        <xsl:variable name="node_content" select="local-name()"/>
        <tr>
          <td class="table_column_uneven">
            <xsl:value-of select="
document($textmodule_path)//type_of_website[@ns:name=$website_id]/type_of_text
modules[@ns:name=tablerow]/specification_of_type[@ns:name=$node_content]/text
module[@ns:language=$language]"/>
          </td>
          <td class="table_column_even">
            <xsl:choose>
              <xsl:when test="count(*)>1">
                <xsl:apply-templates/>
              </xsl:when>
              <xsl:otherwise>
                <xsl:value-of select="/*"/>
              </xsl:otherwise>
            </xsl:choose>
          </td>
        </tr>
      </xsl:for-each>
    </table>
  </div>
</xsl:template>
```

Abbildung 9-7: Tabellenzelleninhalte dynamisch mit Textmodulen befüllen

Quelle: Codeausschnitt aus XSLT-Stylesheet *technical_data.xsl*

Für den Webseitenabschnitt zu den Voraussetzungen, die die vorhandene Software erfüllen muss, um die angebotene Projektsoftware nutzen zu können, wird eine Tabelle angelegt. Der Abschnitt wird durch das Element *software_requirements* vertreten. Für jedes Kindelement des Elementes wird eine Tabellenzeile angelegt. In der ersten Tabellenzelle einer Zeile wird der entsprechende Textbaustein zur Tabellenzeilenbeschriftung ausgelesen. Dies regelt ein XPath-

Ausdruck, der auf die Textmoduldatei zugreift. Den Dateipfad stellt die globale Variable *textmodule_path* bereit. Um auf das benötigte Textmodul innerhalb der Textmoduldatei zuzugreifen, wird die ID des Webseitentyps über die globale Variable *website_id* angegeben. Der Textmodultyp *tabelrow* wird hingegen als String fest eingetragen. Um dem aktuellen Elementknoten eines Schleifendurchgangs den passenden Textmoduleintrag zuzuweisen, wird der Wert der lokalen Variable *node_content* benötigt. Ihr wird bei jedem Schleifendurchgang der Name des aktuellen Elementknotens als Wert zugewiesen. Voraussetzung ist die bereits aufgezeigte Spezifizierung des Textmodultyps in der XML-Textmoduldatei durch den Namen des Elementes, zu dem der Textbaustein gehört.

Automatisches Befüllen von Elementen

Der Einsatz des Verfahrens in einem Template für mehrere Elemente zeigt sich im folgenden modularen XSLT-Stylesheet *contacts_generate*:

```
<!--E-Mail-Adresse | Telefon | Fax mit automatischen Text und
Sprachunterscheidung*****-->
<xsl:template match="email | phone | fax">
  <xsl:variable name="node_content" select="local-name(.)"/>
  <xsl:message>E-Mail | Phone | Fax </xsl:message>
  <tr>
    <td class="website table general column1">
      <xsl:value-of select="
document($textmodule_path)//type_of_website[@ns:name='shared']/type_of_textmodule
s[@ns:name='tabelrow']/specification_of_type[@ns:name=$node_content]/textmodule[
@ns:language=$language]"/>
    </td>
    <td class="website_table_general_column2">
      <xsl:choose>
        <xsl:when test="local-name()='email'">
          <xsl:call-template name="email_generate">
            <xsl:with-param name="Email" select="normalize-space(.)"/>
            <xsl:with-param name="style" select=""/>
          </xsl:call-template>
        </xsl:when>
        <xsl:otherwise>
          <xsl:value-of select="."/>
        </xsl:otherwise>
      </xsl:choose>
    </td>
  </tr>
</xsl:template>
```

Abbildung 9-8: Auslesen von Textmodulen in mehrfach verwendetem Template

Quelle: Codeausschnitt aus XSLT-Stylesheet *contacts_generate.xsl*

Für die drei Elemente *email*, *phone* und *fax* befinden sich im Gegensatz zum vorherigen Beispiel die Tabellenzeilenbeschriftungen in der Textmoduldatei im Abschnitt *shared*, da sie auch für andere Webseiten gelten. Neben *tabelrow* wird im XPath-Ausdruck auch *shared* als String fest eingetragen.

9.5.2 Automatisch generierte Webseite: Aktuelle Projekte

Folgende Abbildung zeigt die Webseite *Aktuelle Projekte*, die in einen Frame eingebunden wird (die Texte zu den Projekten sind von der Autorin beispielhaft eingefügt worden):

Actual projects and closed projects of the actual and the last year



In the following you get an overview of the actual projects and also of the closed projects of this year an the last year:

Actual projects 2007

Name of project	Objective	Developer
→ Grid Techniques	Using Grid-Computing with simple techniques	■ Sven Haug ■ Peter Müller
→ USCT Image Reconstruction with Compressive Sensing	Analyze the appliance aplayments of compressive sensing to USCT imgae reconstruction	Alexander Frank

Finished projects of 2007

Name of project	Objective	Developer
→ GT4Tray	Using Grid-Computing quickly and efficiently without previous knowledge	Thomas Jeijkal
→ GridLJ	A dynamic Grid Service Architecture for scientific image processing	Alexander Frank
→ Reconstruction Omp Grid	Reach one's target as fast as possible	

Finished projects of 2006

Name of project	Objective	Developer
→ Comissar Computing	Find identical objectives	
→ Grail	Grid Access and Instrumentation Tool	Thomas Jeijkal

2007-09-01

Institute for Data Processing and Electronics, Forschungszentrum Karlsruhe

The photo next to the headline from
pixelio.de
Website: <http://www.pixelio.de>

Abbildung 9-9: Webseite *Aktuelle Projekte*

Quelle: Eigene Darstellung

Generierung von Webseite: *Aktuelle Projekte*

Diese Webseite wird per Buttonklick automatisch generiert, wie bereits in Kapitel 6.3.6 aufgeführt. Nachdem alle Projekte in der XML-Datei aufgelistet sind, werden mithilfe von XSLT die Webseitenstruktur sowie die Inhalte erzeugt. Dies bewirkt das Zusammenspiel der verschiedenen bisher genannten Mechanismen: das Auslesen von Inhalten aus den XML-Dateien anderer Webseiten, das Einfügen von Textmodulen aus der zentralen Textmoduldatei sowie die Wiederverwendung der modularen Stylesheets. Die Stylesheetdatei benötigt lediglich das aktuelle Datum, das ihr als Parameter vom *Authentic* übergeben wird.

Für die Unterscheidung der Projekte in laufende Projekte sowie abgeschlossene Projekte des aktuellen und letzten Jahres werden Keys generiert. Die aktuellen Projekte werden als Tabelle ausgegeben und nach Datum absteigend sortiert. Textmodule für den Tabellenkopf werden aus der Textmoduldatei ausgelesen. Für jedes Projekt wird eine Tabellenzeile generiert. Zur besseren Unterscheidung

werden die Tabellenzeilen mit zwei unterschiedlichen Hintergrundfarben ausgegeben. Projektname, Slogan sowie Projektbeteiligte, die in der Übersichtsseite der Veröffentlichung ihres Namens zugestimmt haben, werden aus der XML-Datei der *Projektübersichtsseite* ausgelesen und in den vorgesehenen Tabellenzeilen ausgegeben. Mithilfe von Abfragemechanismen werden ab zwei Entwicklern die Namen als Liste angezeigt, bei einem hingegen nicht. Für jedes Projekt werden zwei Links generiert: Das Pfeilsymbol sowie der Projektname sind bei den aktuellen Projekten auf die Übersichtsseite verlinkt, die bei Projektbeginn alleine das Projekt im Webauftritt vertritt. Entsprechende Tooltips verweisen auf das Linkziel. Analog zu den aktuellen Projekten werden die abgeschlossenen des aktuellen und letzten Jahres generiert. Einziger Unterschied besteht im Linkziel. Die abgeschlossenen Projekte werden auf die Projektstartseite verlinkt. Zwischen den Projektblöcken werden Unterüberschriften mit der entsprechenden Jahreszahl ausgegeben. Die Jahreszahl wird vom übergebenen Datum entsprechend abgeleitet.

9.5.3 Automatisch generierte Webseiten: Startseite und Projektstartseite

Projekt GridIJ

Grid schnell und effizient ohne Vorwissen nutzen

Kurzbeschreibung

Um was geht es?
GRID IJ ist eine Softwareumgebung, die es ermöglicht, Funktionalität einfach auf ein [Grid](#) zu bringen

Auf einen Blick

Das Wichtigste in Kürze
Erfahren Sie knapp Wichtiges zum Projekthalt und den Projektdaten... [→](#)

Leistungsspektrum der Softwarearchitektur

Was zeichnet das Projekt aus?
Erhalten Sie einen Überblick zum Leistungsspektrum des Projektes... [→](#)

Struktur

Aus was setzt es sich zusammen?
Erhalten Sie einen Überblick zum Aufbau der Software... [→](#)

Technische Daten

Was sollte man wissen?
Informieren Sie sich schnell über die Technischen Daten zur Software... [→](#)

Download

Bereit zum Testen?
Hier erhalten Sie die Software und erfahren alles zu den Lizenzen... [→](#)

1111-11-11

Institut für Prozessdatenverarbeitung und Elektronik, Forschungszentrum Karlsruhe

Alle Webseitenphotos von [pixello.de](http://www.pixello.de),
außer Photo von Kurzbeschreibung
Webseite: <http://www.pixello.de>

Abbildung 9-10: Webseite *Projektartseite* für Projekt GridIJ

Quelle: Eigene Darstellung

Generierung von
Webseiten:
Startseite und
Projektstartseite

Die beiden Webseitentypen Startseite und Projektstartseite werden über dasselbe Stylesheet automatisch generiert. Für die Unterscheidung, welche Webseite generiert werden soll, wird als Parameter vom *Authentic* die ID des Webseitentyps übergeben. Analog zur Webseite *Aktuelle Projekte* kommen Textmoduldatei,

Auslesen von Inhalten aus der XML-Datei der Übersichtsseite sowie die Wiederverwendung der modularen Stylesheets zum Einsatz.

Für jeden Teaser, der in der XML-Datei mit *true* als 'zu veröffentlichen' markiert ist, wird eine Tabelle generiert. Die Anordnung der Teaser zueinander regeln *div*-Tags und entsprechende Stilangaben. Je nach Position erstrecken sich die Teaser über die gesamte Satzbreite oder über die Hälfte. Da die Teaseranzahl variieren kann, muss unter anderem überprüft werden, ob der letzte Teaser alleine steht, um für eine kompakte Darstellung den Teaser über die ganze Satzbreite darzustellen.

Für jede Tabelle werden die verschiedenen Texte aus den entsprechenden Abschnitten in der Textmoduldatei ausgelesen. Zudem enthält jeder Teaser als Vertreter einer Webseite das entsprechende Bild des Webseitentyps. Dieses und das Pfeilsymbol sind auf die entsprechende Webseite verlinkt. Tooltips weisen den User auf das Linkziel hin.

9.6 Realisierungsbeispiele für Linktypen

9.6.1 Link auf Glossartermdatei

Wie im Kapitel 6.3.2 bereits vorgestellt, werden die Links auf Glossartermdateien in den XML-Dateien automatisch gesetzt. Für den Linkmechanismus sind die XSLT-Datei *glossaryterm_topic_link* und die JavaScript-Datei *generate_glossaryterm_window* zuständig.

Generierung von Link
glossary-
term_topic_link

Die XSLT-Datei verlinkt auf die HTML Datei des entsprechenden Glossarterms und ruft die JavaScript-Funktion *WinOpen* mit Parameterübergabe auf.

```
<!-- Erstellen des Links mit Linksymbol und JavaScript-Aufruf für das Fenster ***** -->
<a href="#" onclick="WinOpen('${termfile_path}');" title="{ $tooltip_glossaryterm_topic_link }">
  </img>
  <xsl:text> </xsl:text><xsl:value-of select="normalize-space(@ns:link_text)"/>
</a>
```

Abbildung 9-11: Setzen des Linkankers

Quelle: Codeausschnitt aus der XSLT-Datei *glossaryterm_topic_link.xsl*

Die JavaScript-Funktion generiert ein Fenster, in dem sich die Glossartermdatei öffnet. Das Fenster lässt sich über einen Button schließen, der im entsprechenden Stylesheet für Glossarterm-Webseiten generiert wird.

9.6.2 Link auf Seitenabschnitt

Die Linkanker der Linkliste werden über die XSLT-Datei *section_link* generiert, während die Marken über die XSLT-Datei *headlines_generate* erzeugt werden.

Generierung von Link *section_link*

Während bei den Links zum Seitenanfang für den Linkmechanismus bei Anker und Marke die IDs automatisch über XSLT-Funktionen erzeugt werden, wird bei den Links auf Seitenabschnitte eine andere Systematik eingesetzt:

Das Linkziel ist stets eine Unterüberschrift, die als ID den Namen des XML-Elementes erhält, das sie repräsentiert. Beinhaltet eine Webseite mehrere gleichnamige Elemente, für die eine Unterüberschrift generiert wird, wird zur Unterscheidung die Positionsnummer des Elementes innerhalb der Knotenmenge der Unterüberschrift-Elemente als Zusatz angegeben. Dies wird z.B. bei der Webseite *Ziele* angewendet, in der jedes Ziel eine Überschrift erhält.

Ob eine Linkliste gesetzt werden soll, wird im jeweiligen Stylesheet der Webseite an der entsprechenden Stelle abgefragt. Stehen zwei oder mehrere Linkziele zur Verfügung, wird der Generierungsmechanismus für Verlinkungen auf Seitenabschnitte aufgerufen.

Ein Link wird in Abhängigkeit vom XML-Element generiert, das in der Webseite durch eine Unterüberschrift vertreten ist. Der Elementname wird als Linkziel eingesetzt und Linktext sowie Tooltip dynamisch aus der Textmoduldatei ausgelesen.

Bei der Linkliste für die Übersichtsseite gibt es zwei Besonderheiten: Zum einen wird zwischen der Kurz- und Langversion der Datei unterschieden, zum anderen wird ein kurzer Abschnitt automatisch eingefügt, der den Softwareentwickler vorstellt. Auch dieser erhält an der richtigen Stelle innerhalb der Linkliste einen Link.

9.6.3 E-Mail Link mit verschlüsselter E-Mail Adresse

Die Generierung des Links mit verschlüsselter E-Mail Adresse im HTML-Code erfolgt mithilfe der beiden Dateien *email_link.xsl* und *unencryption_email.js* (siehe Anhang K)

Generierung von Link *email_link*

Die vom User angegebene E-Mail Adresse wird aufgeteilt, um Punkte und @-Zeichen durch Bilder zu ersetzen. Die Verschlüsselung der E-Mail Adresse findet ebenfalls in der XSLT-Datei über eine XPath-Funktion statt. Zudem wird beim Link der Aufruf einer JavaScript-Funktion mit Übergabe der verschlüsselten E-Mail eingefügt. Die Funktion ist für die Entschlüsselung der E-Mail Adresse zuständig. Sie hält den passenden „Gegenschlüssel“ bereit. Bei Betätigung des E-Mail Links öffnet sich dadurch das Outlook-Fenster mit der korrekten E-Mail Adresse.

9.7 Realisierung eines zentralen Layouts

Das Layout wird einzig durch eine separate CSS-Datei geregelt. Eine Kombination aus direkter Formatierung im HTML-Code mit Stilangaben im Cascading Stylesheet bewirkt bei Änderungen eine unnötige Suche nach der gewünschten Stelle. Durch die Referenzierung des Stylesheets in die PHP-/ HTML-Webseiten wirken sich außerdem die Stiländerungen direkt auf die Webseiten aus, ohne diese erneut aus XML mithilfe von XSLT transformieren zu müssen.

Die einzelnen IDs und Klassen der CSS-Datei besitzen „sprechende“ Namen, die konsistent benannt sind. Anstelle einer Seitenstrukturierung mithilfe von Tabellen werden *div*-Tags eingesetzt und entsprechend zueinander in Beziehung gesetzt. So werden z. B. mithilfe der CSS-Eigenschaft *float* die *div*-Container nebeneinander platziert.

9.8 Beispielumsetzung von Webseitentypen mit SPS-Stylesheets

Um die XML-Webseiten im *Authentic* bearbeiten zu können, bedarf es einer speziellen Stylesheetart: dem SPS-Stylesheet. Dieses ist an zwei Webseiten exemplarisch umgesetzt worden.

Durch den Einsatz von Textfeldern, Bildern, Auswahllisten, Kontrollmechanismen und Anweisungen an den Autor in Form von Tooltips oder Fließtext wird der User durch ein solches Stylesheet bei der Erstellung seines Inhaltes hilfreich unterstützt.

9.9 Realisierung einer *Authentic*-Projektdatei

Die Projektdatei zeigt die Projektwebseiten im *Authentic* in einer Verzeichnisstruktur an. Mit ihrer Hilfe kann der User sein Projekt im *Authentic* verwalten. Folgendes Bild zeigt die Baumansicht, die durch die SPP-Projekttemplatedatei, im *Authentic* dargestellt wird:

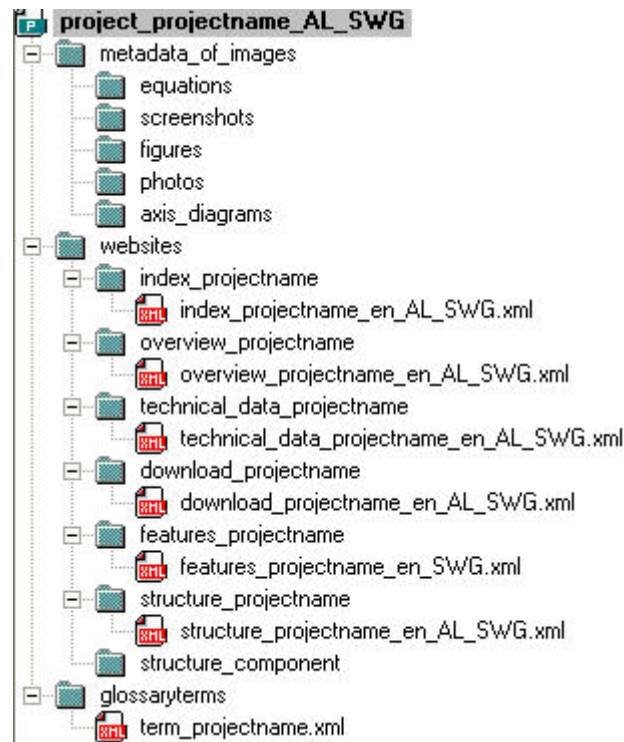


Abbildung 9-12: SPP-Projekttemplatedatei für Webseiten eines Softwareprojektes

Quelle: Eigene Darstellung

Das Anlegen der Datei erfolgt direkt im *Authentic* über die Menüsteuerung. Beim Anlegen eines neuen Projektes wird die Projekttemplatedatei kopiert und entsprechend angepasst (siehe Kapitel 6.3.1).

10 Redaktionsleitfaden

Bei der Erstellung eines kurzen Redaktionsleitfadens für die Grid-Softwaregruppe war das zentrale Ziel einen Leitfaden zu erstellen, mit dem Mitarbeiter und Studenten sich schnell zurecht finden, um ihre Projekte zu dokumentieren (siehe CD Datei *Redaktionsleitfaden.pdf*).

Zunächst werden Zweck und Ziel des Leitfadens dargestellt. Anschließend wird dem Leser erläutert, wie der Leitfaden zu verwenden ist. Den einzelnen Rollen werden ihre Aufgabenpakete zugeteilt. Basiswissen in Form von Beschreibungen und elementaren Handlungsschritten ist in ein separates Kapitel ausgelagert. Dasselbe gilt für den Webauftritt. Hier spiegelt sich der Single Source Gedanke wider. Inhalte, die von mehreren Rollen benötigt werden, müssen nicht mehrfach erklärt werden, sondern sind zentral in einem Kapitel abgelegt.

Quelle für den Redaktionsleitfaden bilden die Exceltabellen zum Webseitenkonzept sowie zu den Automatisierungsprozessen. Aufgrund des engen Zeitrahmens bildete diese Vorgehensweise eine effiziente und schnelle Lösung.

11 Zusammenfassung und Ausblick

11.1 Zusammenfassung der Ergebnisse

Durch die Bearbeitung der breit gefächerten Themengebiete Technologie-Architektur, Prozessdefinition mit Automatisierung, Ablagesystematik, Webauftrittskonzeption mit technischer Realisierung sowie Redaktionsleitfaden gibt die Master-These der Grid-Softwaregruppe eine Komplettlösung an die Hand, mit der ihr Webauftritt standardisiert erzeugt wird.

Technologie-Architektur

Die Technologie-Architektur nutzt bereits vorhandene Technologien und fehlende Komponenten sind durch kostenlose Tools ergänzt. Dies entspricht dem Wunsch der Grid-Softwaregruppe, sich so wenig wie möglich in neue Tools einarbeiten zu müssen. Das Kerntool – der Editor *Authentic*, der zur Bearbeitung der Webseiten in XML eingesetzt wird – erlaubt mit entsprechenden Stylesheets die XML-Dokumente ohne Vorwissen in XML leicht mit Inhalt zu füllen.

Prozessdefinition/Automatisierung

Um effiziente redaktionelle Prozesse zu erhalten, wurden Rollen, Aufgaben und Abläufe definiert. Schwerpunkt bildete dabei die Analyse, welche Prozesse automatisiert durchgeführt werden können. Ein Zusammenspiel von Batch-Programmierung, *Authentic*-Konfiguration und XSLT-Stylesheets realisiert die im Konzept definierten automatisierten Prozesse. Diese erleichtern der Grid-Softwaregruppe ihre redaktionelle Arbeit und unterstützen sie dabei.

Ablagesystematik

Eine Ablagesystematik regelt, wo die bei der Webauftrittserstellung benötigten und anfallenden Dateien und Ordner gespeichert werden. Ergänzend sorgen Dateibenennungsregelungen für eine konsistente Namensgebung. Die zentrale und standardisierte Ablage ermöglicht einen schnellen Zugriff auf die benötigten Inhalte für den Webauftritt.

Webauftrittskonzeption

Die Webauftrittsstruktur ermöglicht es, jedes Softwareprojekt in mehreren Entwicklungsphasen zu präsentieren und in den Gesamtwebauftritt der Grid-Softwaregruppe zu integrieren. Erforderliche und optionale Webseiten sowie eine besondere Gestaltung der Projektstartseite sorgen dafür, dass der Internetuser ab Beginn eines Softwareprojektes die wichtigsten Informationen erhält und alle Projekte mit der gleichen Informationstiefe vertreten sind. Zudem präsentieren sich die Projekte in einem einheitlichen Erscheinungsbild, das die Corporate Identity des Forschungszentrums weiterführt.

Technische Realisierung

Schemas und XML-Templatedateien mit Beispieltextrn sichern die konsistente Sequenzierung der geforderten Inhalte auf den einzelnen Webseiten und erleichtern den Autoren die Arbeit. Unterstützt wird dies durch automatisch eingefügte Textbausteine.

Bei der Konzeption und Realisierung der Webseiten wurde darauf geachtet, dass der Webauftritt leicht erweitert werden kann. Durch die Modularisierung der Schemas und XSLT-Stylesheets lassen sich neue Schemas und Stylesheets aus den bereits vorhandenen größtenteils zusammenbauen.

Der Webauftritt präsentiert sich zunächst nur in der englischen Sprache. Die deutsche Variante ist jedoch bereits realisiert. Für weitere Sprachen lässt sich der Webauftritt leicht erweitern. Die Umsetzung des deutschen Webauftrittes zeigt auch der folgende Vorher-Nachher-Vergleich der Webauftrittstartseite der Grid-Softwaregruppe, die in eine Frameseite eingebunden wird:

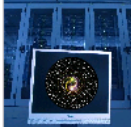
Scientific Software	
Our research topics:	
Grid Computing	
Scientific contributions	Projects
1. Roadmap 2. Grid Services Toolkit 3. GRAIL 4. GT4Tray 5. GridIJ	1. D-Grid 2. AugerAccess 3. EGEE

Abbildung 11-1: Ausschnitt aus der alten Startseite der Grid-Softwaregruppe

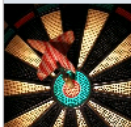
Quelle: <http://fuzzy.fzk.de/%7EGRID/> [Zugriff: 05.09.2007 14:45 MESZ]

Grid-Projekte der Softwaregruppe

Grid - zukunftsweisende Entwicklung



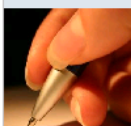
Speicher ohne Grenzen
Grid Computing stellt primär nicht genutzte IT-Ressourcen, die weltweit verteilt sein können, wie Rechenleistung, Speicherplatz, Anwendungen oder Netzwerke als „ganzheitliche Ressource“ zur Verfügung. IT-Betriebsmittel werden so leicht zugänglich und verfügbar gemacht, wie Strom aus einer Steckdose.

Ziele - was wollen wir erreichen?


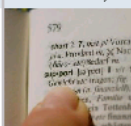
Dynamische Servicearchitektur
Erfahren Sie, welche Zielsetzungen die Basis unserer Projekte bilden ... →

Projekte - woran arbeiten wir?


Alles auf einen Blick
Erhalten Sie einen Überblick von abgeschlossenen und aktuellen Grid-Forschungsprojekte ... →

Publikationen - was gibt es zu berichten?


Erkenntnisse
Erhalten Sie einen Überblick von unseren Publikationen der letzten Jahre ... →

Glossar - Wer, Was, Wie, Warum?


Kurz und bündig
Informieren Sie sich schnell und gezielt über Begriffe zum Thema Grid ... →

Kontakt - Der Weg zu uns



So erreichen Sie uns ...
Kontaktieren Sie uns, wenn Sie Fragen oder Anregungen haben ... →

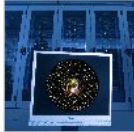
2007-09-01
Institut für Prozessdatenverarbeitung und Elektronik, Forschungszentrum Karlsruhe

Alle Webseitenphotos von pixelio.de,
außer Photo von Kurzbeschreibung
Webseite: <http://www.pixelio.de>

Abbildung 11-2: Neue deutsche Startseite der Grid-Softwaregruppe

Quelle: Eigene Darstellung

Grid-projects of the group of Software Methods

Grid - the future technology  Memory without any limits The idea of Grid computing is to provide worldwide information and computer resources in so far unimaginable quantities. Grid technologies are suited to meet the expected computing requirements of new scientific challenges.	
Objectives - where do we want to go tomorrow?  Dynamical service architecture Come to know, which objectives build the basis of our projects ... →	Projects - what are we working on?  At a glance Get an overview of our closed and actual Grid research projects ... →
Publications - what's important to say?  New knowledge Get an overview about our publications of the last years ... →	Glossary - what does it mean?  In a few words Read up on terms of the topic Grid quickly and specifically ... →
Contact - How to reach us  How to reach us... Contact us, if you have questions or proposals ... →	

2007-09-01
 Institute for Data Processing and Electronics, Forschungszentrum Karlsruhe

All website-photos from [pixello.de](http://www.pixello.de),
 except for short description-photo
 Website: <http://www.pixello.de>

Abbildung 11-3: Neue englische Startseite der Grid-Softwaregruppe

Quelle: Eigene Darstellung

Redaktionsleitfaden

Die definierten redaktionellen Prozesse sind im Redaktionsleitfaden dokumentiert. Die Softwareentwickler erhalten in ihrer Rolle als Redakteur mit ihm einen „Führer“, der sie während der Erstellung des Webauftrittes begleitet.

11.2 Ausblick

Bei der Konzeption und technischen Umsetzung des Webauftrittes spielte der Gedanke des Single Source Publishing eine wichtige Rolle. Einen ersten Schritt in Richtung Cross Media Publishing verwirklicht der Automatisierungsprozess *Formel im TeX-Format in Bild im PNG-Format umwandeln*. Er erzeugt nicht nur aus einer TeX-Datei ein Bild für Webseiten, sondern auch für eine PowerPoint Präsentation. Die Wahl des Datenformates XML bietet weitere Möglichkeiten die Inhalte für andere Medien nutzbar zu machen.

Abschließend seien folgende Anregungen genannt:

- Ein Flyer, der ein Softwareprojekt oder mehrere Projekte präsentiert, kann auf Basis der XML-Projektüberblicksseiten mithilfe entsprechender XSL-Transformationen erstellt werden. Hierfür werden die folgenden Inhalte der Webseiten benötigt: Slogan, Kurzbeschreibung, Motivation und Vorteile sowie Ansprechpartner.

- Um einen Ausgangstext für einen kurzen Werbeartikel im Kundennewsletter *Research to business* des Forschungszentrums zu erhalten, werden dieselben Inhalte wie für den Flyer benötigt.
- Ein Technisches Datenblatt lässt sich aus der Webseite *Technische Daten* durch entsprechende XSL-Transformationen generieren.

Die weitere Umsetzung von Cross Media Publishing würde damit eine effiziente Nutzung der bereits vorhandenen Dokumentationsressourcen in Form von XML-Dokumenten bedeuten.

Literaturverzeichnis

Selbstständige Quellen

- Bartel, Torsten (2004): Die Verbesserung der Usability von WebSites : auf der Basis von WebStyleguides, Usability Testing und Logfile-Analysen. 2. aktual. Aufl. Stuttgart : WiKu
- Closs, Sissi (2007): Single Source Publishing : Topicorientierte Strukturierung und DITA. Unterhaching : entwickler.press
- Fritz, Michael / Straub, Daniela / Ziegler, Wolfgang (2005): Effizientes Informationsmanagement durch spezielle Content-Management-Systeme : Praxishilfe und Leitfaden zu Grundlagen-Auswahl und Einführung-Systemen am Markt Stuttgart : TC and more
- Gaß, Heinz-Peter (1991): Batchprogrammierung unter DOS : ein Praxisbuch für das Erstellen von Batchprogrammen, Grundlagen, Befehle, Techniken, Beispiele. München : IWT
- Grotenhoff Maria, Stylianakis Anna (2002): Website-Konzeption : Von der Idee zum Storyboard. Bonn : Galileo Press
- Grünwied, Gertrud (2007): Software-Dokumentation : Grundlagen-Praxis-Lösungen. Mit 111 Bildern. 2. aktual. Aufl. Renningen : expert (Kontakt & Studium Bd. 668)
- Hennig, Jörg / Tjarks-Sobhani Marita (1998): Wörterbuch zur technischen Kommunikation und Dokumentation. Lübeck : Schmidt-Römhild
- Hoffmann, Walter / Hölscher, Brigitte G. / Thiele, Ulrich (2002): Handbuch für technische Autoren und Redakteure : Produktinformation und Dokumentation im Multimedia-Zeitalter. Erlangen : VDE
- Knobloch, Manfred / Kopp, Matthias (2001): Web-Design mit XML : Webseiten erstellen mit XML, XSL und Cascading Style Sheets. Heidelberg : dpunkt
- Kränzler, Christine (2006): XML/XSL für Buch und Web. 6.Aufl. München: Markt + Technik
- Lehner, Franz (1994): Software-Dokumentation und Messung der Dokumentationsqualität. Mit 55 Abbildungen. München : Carl Hanser
- Lenz, Evan (2006): XSLT 1.0 kurz und gut. Übersetzung von Lars Schulten. Köln : O'Reilly
- Lubkowitz, Mark (2005): Webseiten programmieren und gestalten. 2.erw.Aufl. Bonn : Galileo Press
- Manhartsberger, Martina / Musil, Sabine (2002): Web Usability : Das Prinzip des Vertrauens. Bonn : Galileo Press

- Oehmig, Peter (2004): Redaktionsleitfaden : Grundstein zur Dokumentation.
Schenkenzell : GFT
- Rockley, Ann / Kostur, Pamela / Manning, Steven (2003): Managing Enterprise
Content : A Unified Content Strategy. Berkeley : New Riders
- Skulusch, Marco / Wiederstein, Marcus (2005): XSLT und XPath für HTML, Text
und XML. Bonn : mitp
- Skulusch, Marco / Wiederstein, Marcus (2005): XSLT 2.0 : Fortgeschrittene An-
wendungen. Bonn : mitp
- Tidwell, Doug (2002): XSLT. Deutsche Übersetzung von Kathrin Lichtenberg und
Olaf Brodacki. Köln : O'Reilley
- Voss, Andreas (2000): Das große PC-Lexikon 2000. 2.Aufl. Düsseldorf : Data
Becker
- Wassermann, Tobias (2006): Versionsmanagement mit Subversion : Installation
Konfiguration Administration. Heidelberg : Mitp
- Wenz, Christian/ Hauser, Tobias (2006): HTML: Webseiten einfach selber ma-
chen. 9.Aufl. München : Markt+Technik

Aufsätze

- Heinssen, Sven / Vogt, Petra (2003): „Einleitung: Usability – darum geht's“. In:
Heinssen, Sven /Vogt, Petra (Hrsg.) (2003): Usability praktisch umsetzen :
Handbuch für Software, Web, Mobile Devices und andere interaktive
Produkte. München : Hanser
- Muthig, Jürgen / Schäflein-Armbruster, Robert (1999): „Funktionsdesign: eine
universelle und flexible Standardisierungstechnik“. In: Wallin Felkner,
Christine / Sturz, Wolfgang (Hrsg.) (1999): Technische Dokumentation –
wirtschaftlich organisieren, systematisch erstellen, kundengerecht gestal-
ten. Augsburg : WEKA, 1995 ff.
- Vogt, Petra (2003): „Web-Usability“. In: Heinssen, Sven / Vogt, Petra (Hrsg.)
(2003): Usability praktisch umsetzen : Handbuch für Software, Web, Mo-
bile Devices und andere interaktive Produkte. München : Hanser
- Wochele, Doris / Kupsch, Rainer: „Fallbeispiel: Einsatz eines Enterprise Content
Management Systems für Verwaltung und Forschung“. In: Forschungs-
zentrum Jülich GmbH(Hrsg.) (2005): Knowledge eXtended : Die Koope-
ration von Wissenschaftlern, Bibliothekaren und IT-Spezialisten : 3. Kon-
ferenz der Zentralbibliothek : 2.-4.-November 2005 : Vorträge und Pos-
ter, Jülich : Forschungszentrum, 43-54(Schriften des Forschungszent-
rums Jülich 14)

Gesetze und Normen

- DIN EN ISO 13407 (November 2000): Benutzerorientierte Gestaltung interaktiver Systeme.
- DIN EN ISO 9241 Teil110 (August 2006): Ergonomie der Mensch-System-Interaktion : Grundsätze der Dialoggestaltung.
- DIN EN ISO 9241 Teil11 (Januar 1999): Ergonomische Anforderungen für Büro-tätigkeiten mit Bildschirmgeräten : Anforderungen an die Gebrauchstauglichkeit; Leitsätze.
- IEEE Standards 1063 (Dezember 2001): IEEE Standard for Software User Documentation.
- ISO/IEC 18019 (Januar 2004): Software and system engineering – Guidelines for the design and preparation of user documentation for application software

Zeitschriftenartikel

- Geiger, Beate (1985): „Aufgabenbezogene Struktur von Software-Dokumentation“. In: tekom nachrichten Jg.8 (1985), Nr.1, 12f
- Kammerer, Matthias / Sieber, Tanja (2006): „Sind Metadaten die besseren Daten?“. In: technische kommunikation Jg.28 (2006), Heft 5, 56
- Krings, Hans (2003): „Gebrauchstauglichkeit von Technischer Dokumentation : Vertrauen ist gut, Testen ist besser“. In: technische kommunikation Jg.25 (2003), Heft 5, 26
- Mare Hassenzahl, Axel Platz , Michael Burmester ,Katrin Lehner (2000): “Hedonic and ergonomic quality aspects determine a software's appeal“. In: Proceedings of the SIGCHI conference on Human factors in computing systems (2000), 201-208
- Ziegler, Wolfgang (2004): „Lohnt sich Content Management?“. In: technische Kommunikation Jg.26 (2004), Nr.6, 19
- Zimmermann, Hansjörg (2001): „Web-Usability – Pflicht oder Kür? Anforderungen, Zielgruppen, menschliche Wahrnehmung“. In: technische kommunikation Jg.23 (2001), Nr.5, 17

Internetquellen

- Altova (2007): „XMLSpy - Modellieren, Editieren, Transformieren & Debuggen von XML-Technologien“.
<http://www.altova.com/de/produkte/xmlspy/xml_editor.html>
[Stand: 2007. Zugriff: 16.05.2007, 17:20Uhr MESZ]

- Altova (2007): „Altova Authentic 2007 Desktop Edition”.
<<http://www.altova.com/manual2007/AuthenticDesktop/>>
[Stand: 2007. Zugriff: 01.06.2007, 13:30Uhr MESZ]
- Beck, Astrid (2000): „Guidelines für Web Design”.
<<http://www.guidesign.de/style1.htm>>
[Stand: 2000. Zugriff: 01.08.2007, 17.55Uhr MESZ]
- Ben-Zin (2002): „LaTeX”.
<<http://de.wikipedia.org/wiki/LaTeX>>
[Stand: Juni 2007. Zugriff: 02.06.2007, 14:45Uhr MESZ]
- Ermer, Thomas / Meyer, Michael (2004): „Unix Werkzeuge - Der Stream Editor”.
<<http://gd.tuwien.ac.at/opsys/linux/tut/LinuxFibel/sed.htm>>
[Stand: Januar 2004. Zugriff: 25.05.2007, 18:30Uhr MESZ]
- ETU (2004): „MathML”.
<<http://en.wikipedia.org/wiki/MathML>>
[Stand: May 2007. Zugriff: 02.06.2007, 11:45Uhr MESZ]
- Greif, Norbert / Saborrosch, Doris (2005): „Richtlinien für die Softwaredokumentation”.
<http://ib.ptb.de/pls/portal/docs/PAGE/IB_PTB/8/85/851/SPS/SWQ/DOC/drl/PTB-8.51-AD01-Doku-V30.pdf>
[Stand: September 2005 Zugriff: 01.05.2007, 15:20 MESZ]
- Guillaume, Damien (2007): „Jaxe, your XML editor”.
<<http://jaxe.sourceforge.net/>>
[Stand: April 2007. Zugriff: 14.05.2007, 11:30Uhr MESZ]
- Jacobsen, Jens (2003): „Metadaten richtig anlegen – Verstecken Sie Ihre Seiten nicht!”.
<http://www.contentmanager.de/magazin/artikel_372_metadaten_richtig_anlegen.html>
[Stand: 2007. Zugriff: 02.08.2007, 18:20 MESZ]
- Küng, Stefan / Onken, Lübke / Large, Simon (2006): „TortoiseSVN :
Ein Subversion-Client für Windows Version 1.4.4”.
<http://tortoisesvn.net/docs/release/TortoiseSVN_de/index.html>
[Stand: November 2007. Zugriff: 14.05.2007, 09:55Uhr MESZ]
- Manhart, Klaus (2003): „XML-Editoren für Windows und Linux”.
<<http://www.tecchannel.de/entwicklung/programmierung/402216/index.html>>
[Stand: Dezember 2003. Zugriff: 30.04.2007, 14:15Uhr MESZ]
- Microsoft Corporation (2007): „XML Notepad 2007”.
<<http://www.microsoft.com/downloads/details.aspx?FamilyID=72d6aa49->

- 787d-4118-ba5f-4f30fe913628&DisplayLang=en>
[Stand: 2007. Zugriff: 21.05.2007, 11:45Uhr MESZ]
- Mozilla.org (1999): „MathML in Mozilla“.
<<http://www.mozilla.org/projects/mathml/>>
[Stand: September 2006. Zugriff: 02.06.2007, 12:00Uhr MESZ]
- Muranko, Beate / Drechsler, Rolf (2005): „Technische Dokumentation von Soft- und Hardware-Systemen: Die vergessene Welt“.
<http://www.informatik.uni-bremen.de/agra/doc/work/GI_ITG_GMM_tDoku_20051.pdf>
[Stand: k.A. Zugriff: 20.04.2007, 10:00Uhr MESZ]
- Nateqi Jama (2003): „Veröffentlichung von Mathematik im Web mit MathML“.
<http://www.matheboard.de/docs/W3C/Math/XSL/putting_math_on_the_web_german.htm>
[Stand: k.A. Zugriff: 02.06.2006, 13:00Uhr MESZ]
- Pixware (2007): „XMLmind XML Editor“.
<<http://www.xmlmind.com/xmleditor/>>
[Stand: 2007. Zugriff: 22.05.2007, 11:00Uhr MESZ]
- Prilmeier, Franz (2006): „Versionskontrolle von Webseiten am Beispiel CVS“.
<<http://aktuell.de.selfhtml.org/artikel/projekt/cvs/>> [Stand: k.A. Zugriff: 12.05.2007, 13:00Uhr MESZ]
- Reuther, Ursula (2006): „Gut für die Struktur“.
<http://www.tekom.de/index_neu.jsp?url=/vollanzeige_modul.jsp>
[Stand: Mai 2006. Zugriff: 15.05.2007, 18:40 MESZ]
- Schulz, Ursula (2006): „Web Usability“.
<<http://www.bui.fh-hamburg.de/pers/ursula.schulz/webusability/webusability.html>>
[Stand: August 2006. Zugriff: 06.08.2007, 09:25 MESZ]
- Schwermer, Rolf (1998): „Redaktionsleitfäden – Ein unersetzliches Hilfsmittel in der Technikredaktion“.
<http://www.doculine.com/news/1998/04_98/Redaktionsleitfaden.htm>
[Stand: Oktober. 2005 Zugriff: 30.08.2007, 13:10 MESZ]
- SyncRO Soft Ltd. (2007): „<oxygen> XML Editor &XSLT Debugger“.
<<http://www.oxygenxml.com/>>
[Stand: 2007. Zugriff: 22.05.2007, 14:00Uhr MESZ]
- Wirth, Thomas (2005): „DIN EN ISO 9241 - 10“.
<www.kommdesign.de/texte/din.htm>
[Stand: Oktober 2005. Zugriff: 02.07.2007, 11:20Uhr MESZ]

Hochschulschriften

- Eberhard, David (1996): Zur Relevanz der Dokumentation und Kommunikation in der betrieblichen Softwareentwicklung und –wartung : Eine Konzeption zur Verbesserung der Effizienz von Dokumentation und Kommunikation in der betrieblichen Softwareentwicklung mittels CSCW-basierter Technologie und der Integration von Hypertextfunktionalitäten zur Informationsvernetzung. Paderborn, Universität-Gesamthochschule, Fachbereich Wirtschaftswissenschaften, Diss., 1996
- Hermann, Fabian (2006): Usability Engineering für die Technische Dokumentation. Hochschule Karlsruhe, Fakultät für Wirtschaftswissenschaften, Studiengang Technische Redaktion, Skript zur Vorlesung Usability Engineering, SS 2006
- Schober, Martin (2004): Projektdefinition zur Veranstaltung TR 0620 (Usability Testing). Hochschule Karlsruhe, Fakultät für Wirtschaftswissenschaften, Studiengang Technische Redaktion, Skript zur Übung Usability Projekt, SS 2004
- Ziegler, Wolfgang (2006): Content Management : Grundlage von Redaktionssystemen. Hochschule Karlsruhe, Fakultät für Wirtschaftswissenschaften, Studiengang Technische Redaktion, Folien zur Vorlesung Content Management Systeme, WS 2006/2007

Weitere Quellen

- Beat, Kirchhofer (2005): „Tools/Werkzeuge zur Unterstützung von Evaluationen – Präferenzmatrix“. In: Fritz, Michael et. al. (2005): CD-Beilage
- , (2005): „Tools/Werkzeuge zur Unterstützung von Evaluationen“. In: Fritz, Michael et. al. (2005): CD-Beilage
- Bohrmann, Silke (26.07.2007 10:36 MESZ): „AW: Aktuelles Stylesheet “.E-Mail an Fabienne Holdermann

Anhang

A. Fragenkatalog zur Ist-Analyse am Beispiel *Autor*

Name:		Status:		
Projekte:				
Erstellungs- und Änderungsprozess				
Anstoßprozess	Details	Erläuterung	Antwort	Intention
Wer oder was gibt bei Ihrem Softwareprojekt den Anstoß mit der Dokumentation zu beginnen?		Zur Codekommentierung zur redaktionellen Dokumentation		Workflow: Erfassungsbeginn
Zu welchem Zeitpunkt der Softwareentwicklung erhalten sie den Anstoß mit der Dokumentation zu beginnen?		Für Codekommentar für redaktionelle Dokumentation		Workflow: Zeitpunkt der Dokumentationserstellung in Bezug zum Status des Softwareprojektes
	Anfang des Softwareprojektes			
	Mitte des Softwareprojektes			
	Ende des Softwareprojektes			
	Durchgängig/ begleitend zum Softwareprojekt			
Auf welchem Wege erhalten Sie die Nachricht mit der Dokumentation zu beginnen?				IT-Architektur: Medium
	Papiernotiz			
	Telefon			
	E-Mail			
	Meeting			
	Sonstiges			
Sind Sie mit dieser Vorgehensweise zufrieden? Was könnte verbessert werden?				Problem: Anstoßprozess der Dokumentation Verbesserungsvorschlag
Dokumentationsarten	Details	Erläuterung	Antwort	Intention
Welche Dokumentationstypen erstellen Sie für Ihre Softwareprojekte?		Beispiele: Handbuch, wissenschaftlicher Bericht, Grafiken, Fotos, Programmiercodekommentare, Filme, Strukturabbildungen, Formeln		Dokumentationstypen pro Softwareprojekt; Anforderungen an das Informationsmodell
Sind diese Dokumentationstypen für ihr Softwareprojekt festgelegt?		Alle Dokumente (Projektplanung bis Anleitung)		Definierte Dokumentationstypen, die pro Softwareprojekt erstellt werden müssen
Wenn ja, von wem werden die Dokumentationstypen festgelegt?				Workflow: Rolle/Aufgabe
Wo sind die Vorgaben zu den Dokumentationstypen festgelegt?				IT-Architektur: Speichermedium
Wenn keine Vorgaben existieren, wie wird entschieden, welche Inhalte für Ihre Dokumentation wichtig sind? Ist dies projektabhängig?				Workflow: Rolle/Aufgabe Abhängigkeit der Dokumentationstypen von der Softwareprojektart
Auf welchem Wege erhalten Sie die Information, welche Inhalte Sie erstellen müssen?				Workflow: Medium

Name:	Status:	
Projekte:		

Erstellungs- und Änderungsprozess

Nutzer der Dokumentation	Details	Erläuterung	Antwort	Intention
Wer sind die internen und externen Nutzergruppen Ihrer Dokumentationstypen?				Nutzergruppen der Dokumentationstypen
Gibt es verschiedene Nutzergruppen pro Dokumentationstyp?				Contentvarianten durch unterschiedliche Nutzergruppen
	Haben die Nutzergruppen pro Dokumentation unterschiedliches Vorwissen und tragen Sie diesem Umstand Rechnung?	z.B. ein Manual oder ein wissenschaftlicher Bericht für Nutzergruppen mit unterschiedlichem Vorwissen		Qualität der Dokumentation Problem: fehlende Berücksichtigung des Vorwissens
	Worin unterscheiden sich die Inhalte einer Dokumentation für die verschiedenen Nutzergruppen?	Unterschiede, wenn ein Inhalt in jeweils abgeänderter Form für verschiedene Nutzer zur Verfügung gestellt wird		Unterschiede der Contentvarianten
Welches Feedback erhalten Sie von den Nutzergruppen, speziell von Nutzern außerhalb der Grid-Softwaregruppe? Kommen Nachfragen seitens der Nutzer zu publizierten Inhalten?				Qualität der Dokumentation Problem: Qualität der Dokumentation
Recherche und Informationslieferanten	Details	Erläuterung	Antwort	Intention
Woher bzw. von wem bekommen Sie Informationen für die Erstellung Ihrer Softwaredokumentation?		Intern und extern		Workflow: Informationsquellen
	Andere Personen der Grid-Softwaregruppe: Welche?			
	Andere Forschergruppen des IPE-Instituts: Welche?			
	Andere Institute des Forschungszentrums: Welche?			
	Externe Forschergruppen (Großprojekte)			
	Weitere Quellen wie Internet, Firmen und Fachzeitschriften, Lexika, Wörterbücher Welche?			
	Sonstiges			
Wie erhalten Sie die Informationen für die Softwaredokumentation?				IT-Struktur: Medium
	E-Mail			
	Mündlich			
	Datenbank			
	Server			

Name:	Status:	
Projekte:		

Erstellungs- und Änderungsprozess

Wie erhalten Sie die Informationen für die Softwaredokumentation?	Management-Systeme/Wikis			IT-Struktur: Medium
	Internetseiten			
	Sonstiges			
Um welche Informationen zur Softwaredokumentation handelt es sich?		Daten, Softwarecode, Grafiken, Formeln		Benötigte Inhalte zur Dokumentation
In welchem Datenformat liegen die erhaltenen bzw. gefundenen Informationen zur Softwaredokumentation vor?		Papier, HTML, Java etc.		IT-Struktur: Datenformat
Bei welchen Inhalten besteht für Sie Holschuld?				Workflow: Rolle/Aufgabe
Stehen die Inhalte rechtzeitig zur Verfügung?				Workflow: Informationsbereitstellung
Wenn nein, warum?				Problem: Verzögerung
Stehen die Inhalte korrekt zur Verfügung?		Keine Arbeit mit veraltetem Informationsstand		Problem: Qualität der Informationen
Wenn nein, warum?				
Wie kommen Sie mit den Inhalten zurecht?				
	Zu detailliert? Zu kurz?			Problem: Qualität der Informationen
	Schwierig zu verstehen? Schlechte Qualität?			
Wie müsste der Inhalt sein, um ihn zu verstehen?				Anforderungen an den Inhalt/-saufbau Informationsmodell
Welchen Herausforderungen/ Problemen sehen Sie sich bei der Recherche nach den benötigten Informationen gegenüber?				Problem: Informationsrecherche
Dokumentationserstellung	Details	Erläuterung	Antwort	Intention
Wie werden die erhaltenen bzw. gefundenen Inhalte weiterverwendet?				
	Werden die Inhalte nur ergänzt, etwas Neues hinzugefügt? Worin bestehen die Ergänzungen?	Manual, Wissenschaftlicher Bericht, Grafiken, Fotos, Programmiercodekommentare, Filme, Strukturabbildungen, Formeln		Wiederverwendung von Inhalten durch Ergänzungen
	Werden bestehende Inhalte geändert? Wie sehen die Änderungen aus?	Manual, Wissenschaftlicher Bericht, Grafiken, Fotos, Programmiercodekommentare, Filme, Strukturabbildungen, Formeln		Wiederverwendung von Inhalten durch Änderungen
	Werden Teile von bestehenden Inhalten im neuen Text exakt wiederverwendet?			Exakte Wiederverwendung von Inhalten

Name:		Status:	
Projekte:			
Erstellungs- und Änderungsprozess			
Wie werden die erhaltenen bzw. gefundenen Inhalte weiterverwendet?	Werden Teile von bestehenden Inhalten im neuen Text in abgeänderter Form wiederverwendet? Wie sehen die Änderungen aus? Inhalt komplett neu erfasst		Wiederverwendung von Teilen von Inhalten durch Änderungen
Welche Herausforderungen ergeben sich bei der Weiterverarbeitung?			Komplett neuer Inhalt Problem: Inhaltsverarbeitung
Über welches Medium/Tool erfassen Sie Ihre Inhalte?		(Schreibmaschine, Stift, PC)	IT-Architektur: Editor zur Inhaltserfassung
Müssen Daten in andere Formate konvertiert werden, um sie zu verwenden? Wenn ja welche, welche Schnittstellen gibt es?			IT-Architektur: Datenkonvertierung
Gibt es Probleme bzw. Verbesserungsbedarf hinsichtlich des Editors?			Problem: Editor zur Inhaltserfassung
In welcher Sprache verfassen Sie die Dokumentation? Von was ist die Ausgangssprache abhängig?		Auftraggebern etc.	Ausgangssprachen der Dokumentation
Wie hoch ist Ihr Arbeitsaufwand für die Dokumentationserstellung?			Problem: fehlende Zeiteffektivität
Wo liegen die Herausforderungen beim Erfassen? Was wäre hilfreich, damit die Erstellung schneller bzw. effektiver verläuft?			Problem: Inhaltserfassung Verbesserungsvorschlag
Verfassen Sie die gesamte Dokumentation oder einen Dokumentationstyp alleine für Ihre Projekte oder mit anderen zusammen? Wenn ja mit wem?			Mehrere Autoren schreiben an der gleichen Dokumentation
Kann es sein, dass sie künftig mit anderen zusammen (Grid-Softwaregruppe, andere Forschergruppen am IPE/extern) die Dokumentation für ein Softwareprojekt schreiben müssen?		Mehrer Autoren an einer Dokumentation Mehrere Autoren an gesamter Softwaredokumentation	Zukünftige Anforderungen: Mehrere Autoren schreiben an der gleichen Dokumentation
Welchen Herausforderungen sehen Sie sich beim Schreiben mit anderen zusammen gegenüber?			Problem: Dokumentation durch mehrere Autoren
Welche Vorgaben zum Aufbau der Dokumentationen gibt es?		Grobstruktur	Standards zur Dokumentstruktur
Woher erfahren Sie die Regelungen zum Dokumentationsaufbau und wo sind die Regelungen festgehalten?			Workflow: Rolle/Aufgabe IT-Architektur: Speichermedium
Gibt es Vorgaben bezüglich der Sprache (Satzbau, Terminologie)?		Kontrollierte Sprache Verwendung von Wörtern	Standards zur Sprache

Name:	Status:			
Projekte:				
Erstellungs- und Änderungsprozess				
Woher erfahren Sie die Sprachvorgaben und wo sind diese festgehalten?				Workflow: Rolle/Aufgabe IT-Architektur: Speichermedium
Verwaltung/Ablage	Details	Erläuterung		Intention
Welche erstellten Inhalte legen Sie wo und wie ab?		Speichermedium: Blatt, Datenbank, Abteilungsserver Kennzeichnung: Ordernamen Daten: Informationen/Grafiken, Fotos, Zeichnungen		IT-Architektur: Speichermedium, Ablagesystem Problem: unterschiedliche Ablagesys-teme
Gibt es Benennungsregeln für die erstellten Inhalte (Datei, Grafik etc.) und wenn nein, wie benennen Sie ihre Inhalte?				Standards zur Dateibenennung Problem: Benennungssystem
Werden die Inhalte versioniert und wenn ja, welche Inhalte?		Nur freigegebene Dokumentation		Versionierung: Dokumentationsarten
	Werden geänderte Inhalte versioniert bzw. unter neuem Namen abgespeichert?			Versionierungsvorgang
	Werden ergänzte Inhalte versioniert bzw. unter neuem Namen abgespeichert?			Versionierungsvorgang
Gibt es Regelungen für die Versionierung von Inhalten?		Dateibenennung für Versionen, die noch nicht freigegeben sind		Standards zur Versionierung
In welchem Datenformat werden die erstellten Inhalte abgespeichert?				IT-Architektur: Datenformat
Wer kann auf die abgelegten Inhalte zugreifen?				Workflow: Zugriffsrechte
Wie oft werden Backups gemacht?				Archivierung: Häufigkeit
Von wem werden Backups gemacht?				Workflow: Aufgabe/Rolle
Welches Speichermedium wird für die Backups verwendet?				IT-Architektur: Speichermedium
Existieren Metainformationen zu den Inhalten, z.B. wer den Inhalt erstellt hat, welchen Status der Inhalt hat (z.B. lektoriert, publiziert)?				Metadaten zur Dokumentationserfassung
Hätten Sie gerne noch weitere Metadaten?				Verbesserungsvorschlag: Metadaten
Wenn keine Metadaten existieren, welche wären Ihrer Meinung nach sinnvoll?				Verbesserungsvorschlag: Metadaten
Weiterleitung an Lektor, Übersetzer, Layouter, Publizieren	Details	Erläuterung	Antwort	Intention
An wen wird die Dokumentation nach der Erstellung weitergeleitet bzw. wozu?				Workflow: Aufgabe/Rolle
Wird das Layout vor oder nach dem Lektorat hinzugefügt?		Anbindung von Templates		Workflow: Layout
Auf welchem Wege (Medium) gelangt die Dokumentation dorthin?				IT-Architektur: Datenformat

Name:	Status:	
Projekte:		

Erstellungs- und Änderungsprozess

Besteht für Sie Bringschuld hinsichtlich der Weiterleitung?				Workflow: Aufgabe/Rolle
Stehen die Texte zur Weitergabe rechtzeitig zur Verfügung? Werden Termine vereinbart?				Workflow: Termineinhaltung Problem: Terminverzögerung
Wenn nicht, wodurch entstehen i.d.R. Verzögerungen?				Problemursache: Verzögerung
Änderungs-, Aktualisierungsprozess	Details	Erläuterung	Antwort	Intention
Was ist der Auslöser für Änderungen Ihrer Dokumentation?				Workflow: Änderungsbeginn
Auf welchem Wege erhalten Sie die Nachricht zur Änderung?		Medium		IT-Architektur: Medium
In welchem Status befindet sich Ihre Dokumentation zu diesem Zeitpunkt?				Dokumentationsstatus bei Änderungen
Zu welchem Zeitpunkt des Softwareprojektes tritt der Änderungsprozess ein?				Workflow: Zeitpunkt der Dokumentationsänderung in Bezug zum Status des Softwareprojektes
Um welche Änderungen handelt es sich?				Änderungsarten
Wie oft werden Aktualisierungen von einer Dokumentation durchgeführt?				Änderungshäufigkeit
Muss mehr als eine Dokumentation geändert werden? Welche Dokumentation?				Dokumentationen, die von der Änderung betroffen sind
Wer muss über die Änderungen der Dokumentation informiert werden?		Bringschuld Informationsweitergabe		Workflow: Aufgabe/Rolle

B. Bewertungsmatrix und Präferenzmatrix am Beispiel *Redaktionelle Anforderungen*

Lösungsvergleich					Lösung			
		Ebene 1	Ebene2	Wert	Punkte 1 - 6	Wert	Beschreibung	Begründung der Relevanz
1. Kosten		10%						
Pflicht-Kriterium	1.1	Keine oder geringe Kosten einer Floating Lizenz	60%	6,00%	0	0	Lizenz ist nicht an Personen gebunden	Ständig wechselnde Studierenden und Mitarbeiter sowie neue PCs
	1.2	Kostenlose oder günstige Updates	40%	4,00%	0	0	Geringe Folgekosten	Wichtig aufgrund der vielen Studenten und PCs
			100%					
2. Technischen Anforderungen		30%						
Pflicht-Kriterium	2.1	Funktioniert auf den beiden Betriebssystemen Windows und Linux	4%	1,07%	0	0	Sowohl Mitarbeiter als auch Studenten arbeiten auf unterschiedlichen Betriebssystemen	Jeder soll an seinem eigenen PC arbeiten können. Einen PC nur für redaktionelle Arbeiten ist aus psychologischer Sicht nicht günstig: „heißer Stuhl“
	2.3	Interner Prozessor für XSLT 2.0 und XSL-FO vorhanden oder Einbinden eines externen Prozessors (Saxon) möglich	7%	2,14%	0	0	Mit dem Prozessor wird das XML-Dokument per Buttonklick bzw. Zuweisung des XSLT-Stylesheets in PHP/HTML transformiert	Alternative: Saxon /Batchaufruf
Pflicht-kriterium	2.4	XML-Dokument wird auf Wohlgeformtheit überprüft	25%	7,50%	0	0	XML-Dokument entspricht den Regelungen des W3C Konsortiums	Valide Dateien sind Voraussetzung für die Transformation mit XSLT
Pflicht-kriterium	2.5	XML-Dokument wird gegen sein Schema validiert	21%	6,42%	0	0	XML-Struktur entspricht den Schemavorgaben	Die XML-Dokumente von jedem Projekt müssen dieselbe Struktur aufweisen, um die Konsistenz der Webseiteninhalte zu sichern, v.a. hinsichtlich der Sequenzierung der Inhalte; valide XML-Dokumente sind Voraussetzung für die Transformation mit XSLT
	2.6	Anbindung an Versionierungssystem SVN vorhanden	4%	1,07%	0	0	Ein Client für SVN ist vorhanden	Tortoise ist eine Alternative für den Client und bereits am IPE im Einsatz
	2.7	Prototyp-Projekte definierbar und wiederverwendbar	18%	5,37%	0	0	Projekt anlegen als Prototyp und dieses für jedes Projekt durch Anpassung wiederverwenden	Der Autor hat all seine XML-Dateien gebündelt in einem Projekt; übersichtlich und userfreundlich
	2.8	Oberfläche ist leicht konfigurierbar	11%	3,21%	0	0	Die Sprache ist einstellbar, Symbole können aus der Taskleiste entfernt werden	Geringere Einarbeitung: Übersichtlichkeit der Oberfläche
Pflicht-kriterium	2.9	Systemprozesse durch Programmierung erweiterbar oder Batch/Shell-Dateien einbindbar	11%	3,21%	0	0	Shell-, Skriptsprache oder gut dokumentierte spezifische Tool-Sprache; Projekten oder Ordern zuweisbar	Automatisierungsprozesse müssen mithilfe des Tools durchführbar sein, um die Arbeit des Users zu erleichtern; dient Fehlerquellen zu reduzieren
			100%					
3. Redaktionelle Anforderungen		40%						
Pflicht-kriterium	3.1	Eingabehilfenfenster für erlaubte Elemente und Attribute	14%	5,44%	0	0	Elemente und Attribute, die an der Stelle erlaubt sind, werden in einem separaten Fenster dem User angezeigt	Fehleranfälligkeit beim Taggen minimieren

Lösungsvergleich						Lösung						
		Ebene 1	Ebene2	Wert	Punkte 1 - 6	Wert	Beschreibung	Begründung der Relevanz				
Pflicht-kriterium	3.2	Informationsfenster zum Element/Attribut am Mauszeiger					9%	3,64%	0	0	Am Mauszeiger erscheint ein Tooltip, der das Element bzw. seinen Inhalt beschreibt.	Der User wird durch die Angaben bei der Eingabe geleitet, die Fehleranfälligkeit wird minimiert.
	3.3	Verständliche Fehlermeldungen					11%	4,24%	0	0	Ursache und Art des Fehlers in verständlicher Form. Gibt z.B. der User einen falschen Wert ein, soll das Tool den Autor auf die Ursache klar hinweisen	Fehlermeldungen sind ein wichtiges Hilfsmittel, bei der Eingabe von Textinhalt, wenn dieser bestimmten Einschränkungen unterliegt
Pflicht-kriterium	3.4	Auto-Completion					9%	3,64%	0	0	Bei XML-Eingabe z.B. Enumeration möglich	Erleichtern dem User die Eingabe und reduzieren die Fehleranfälligkeit
Pflicht-kriterium	3.5	Generierung des Outputs mit wenigen Klicks					5%	1,82%	0	0	Die PHP/HTML-Dateien eines Ordners werden auf einmal generiert oder mit wenig Aufwand	Einfache Bedienung, userfreundlich; besonders, wenn Dateien geändert und schnell generiert werden müssen (z.B. vor Messen).
Pflicht-kriterium	3.6	Online-Validierung					17%	6,68%	0	0	Bei der Eingabe wird sofort überprüft, ob das Element gesetzt werden darf oder schon vorher vorhanden ist und an der Stelle nicht erlaubt ist.	Fehlermeldungen sollten aufgrund der Hilfestellungen bei der Eingabe ausgeschlossen sein, was die Sequenzierung der Elemente betrifft.
	3.7	WYSIWYG-Ansicht					2%	0,61%	0	0	Zeigt dem User wie die Output-Datei aussieht	Motivation für den User
Pflicht-kriterium	3.8	Formularansicht					15%	6,08%	0	0	Der User kann in Feldern Text eingeben oder über Radiobuttons und Auswahllisten Werte auswählen	Leichte Eingabe für den User, gute Benutzerführung
	3.9	Baumansicht					2%	0,61%	0	0	Die Struktur bzw. Hierarchie der XML-Elementverschachtelungen ist visualisiert	Erlaubt dem User schnell zu erkennen, wo er sich in der Dokumentstruktur befindet
	3.10	Selbsterklärende Icons					3%	1,21%	0	0	Die Icons sind leicht verständlich	Der User findet sich leicht zurecht
	3.11	Übersichtliche Oberfläche					9%	3,64%	0	0	Wenige Symbole, klarer übersichtlicher Aufbau der Oberfläche	Der User findet sich leicht zurecht
	3.12	Leichte Navigation					6%	2,42%	0	0	Der User soll sich im Editor leicht zurechtfinden können selbsterklärende Menüeinträge	Der User findet sich leicht zurecht
							100%					
4. Nicht funktionale Anforderungen		20%										
	4.1	Tool ist bereits vorhanden					20%	4,00%	0	0	Tool muss nicht neu angeschafft werden	Keine neuen Kosten
	4.2	Mitarbeiter sind mit dem Tool bereits vertraut					50%	10,00%	0	0	Keine Einarbeitung nötig; Wissen bereits vorhanden	Das Tool ist sofort einsetzbar; die Mitarbeiter müssen sich nicht neu einarbeiten
	4.3	Leichte Einarbeitung für Autoren und Administrator					30%	6,00%	0	0	Kurze Einarbeitszeit	Erhöht die Akzeptanz bei den Mitarbeitern ; schneller produktiver Einsatz des Tools möglich
							100%					
						Total	100%	100,01%		0		

Präferenzmatrix				
für Vergleich der		XML-Editoren		
Gewicht	Rangfolge	Anzahl Nennungen	Kriterien	
13,64	3	9	a	Eingabehilfenfenster für erlaubte Elementen und Attribute
9,091	5	6	b	Informationsfenster zum Element/Attribut am Mauszeiger
10,61	4	7	c	Verständliche Fehlermeldungen
9,091	5	6	d	Auto-Completion
4,545	9	3	e	Generierung des Ouputs mit wenigen Klicks
16,67	1	11	f	Online-Validierung
1,515	11	1	g	WYSIWYG-Ansicht
15,15	2	10	h	Formularansicht
1,515	11	1	i	Baumansicht
3,03	10	2	j	Selbsterklärende Icons
9,091	5	6	k	Übersichtliche Oberfläche
6,061	8	4	l	Leichte Navigation
0	13	0	m	
0	13	0	n	
0	13	0	o	
0	13	0	p	
0	13	0	q	
0	13	0	r	
0	13	0	s	
0	13	0	t	
Total		100	66	

C. Bewertungsmatrix für Formel in Bild Umwandlung und eine bewertete Lösung

Lösungsvergleich					Lösung x			
		Ebene 1	Ebene2	Wert	Punkte 1 - 6	Wert	Beschreibung	Begründung der Relevanz
1. Installationsaufwand		15%						
1.1	Geringer Installationsaufwand		100%	15,00%	0	0	Es soll möglichst wenig neue Software auf den Rechnern zu installieren sein	Viele, wechselnde Rechner
			100%					
2. Redaktionelle Tätigkeiten		50%						
2.1	Wenige Schritte zur Durchführung		10%	4,76%	0	0	Von der Formelerstellung bis zur Einbindung in die Webseite sollen wenige Schritte nötig sein	Userfreundlich, besseres Annehmen der Lösung seitens der User aufgrund geringen Aufwandes und kürzerer Erstellungszeiten
2.2	Automatisierungsgrad hoch (wenig manuell)		14%	7,15%	0	0	Diese Schritte sollen so oft wie möglich nicht vom Autor selbst, sondern automatisch ohne sein Zutun durchgeführt werden	Reduzierung der Fehlerquote, z.B. durch Abspeichern im falschen Ordner, Reduzierung des Zeitaufwands für die Erstellung und damit auch besseres Annehmen der Lösung seitens der User
2.3	User kann bekannte Tools verwenden		19%	9,50%	0	0	Der Prozess soll so viel wie möglich mithilfe bereits bekannter Tools durchgeführt werden	Besseres Annehmen der Lösung seitens der User, da vertraut im Umgang mit den Tools
2.4	User muss zur Durchführung wenig neue Kenntnisse erwerben		24%	11,90%	0	0	User soll sich möglichst wenig neues Wissen aneignen müssen, um den Prozess durchführen zu können	Die User haben keine Zeit, neue Programmiersprachen zu lernen; Besseres Annehmen der Lösung seitens der User, wenn ohne großen Aufwand durchführbar
2.5	Formel in WYSIWYG-Ansicht vom XML-Editor anschaubar		10%	4,76%	0	0	Die Formeln sollen in der WYSIWYG-Ansicht angezeigt werden	User hat Eindruck, wie Formel in der Webseite aussieht; Überprüfung, ob der Dateipfad stimmt, da Formel in WYSIWYG sonst nicht anzeigbar; damit userfreundlich und Fehlerreduzierung
2.6	Änderungen werden zentral, an einer Stelle getätigt		5%	2,38%	0	0	Autor oder Reviewer sollen bestehende Formeln nur an einer Stelle (Quelldatei) ändern müssen	Zeitreduzierung; Qualitätssteigerung, da keine Quelle übersehen wird

Lösungsvergleich						Lösung x		Beschreibung	Begründung der Relevanz
						Punkte 1 - 6	Wert		
2.7	Aktualisierungsaufwand nach Änderungen ist gering					0	0	Die Aktualisierung aller betroffenen Dateien soll automatisiert oder mit möglichst wenigen, leichten Schritten durchgeführt werden können	Aufwand- und Zeitreduzierung, da User keine/wenige Seite/n neu generieren und hochladen muss; Qualitätssteigerung, da die Gefahr reduziert ist, zu aktualisierende Seiten zu übersehen
			100%						
3. Userfreundliche Webseiten						30%			
3.1	Schriftgröße der Formel einstellbar					0	0	Die Schriftgröße der Formel soll identisch sein mit der Schriftgröße des Fließtextes	Formel soll optisch zur Webseite passen („harmonisches Gesamtbild“), ansonsten wirkt die Seite unprofessionell und nicht durchdacht; Präsentation nach außen wichtig
3.2	Barrierefreiheit/Keine zusätzliche Installation für User							User soll ohne zusätzliche Plugins bzw. zusätzliche Software installieren zu müssen, die Formeln in seinem Browser anschauen können	Nicht angezeigte Formeln bedeuten unprofessionelle Seiten, schlechtes Image für Grid-Softwaregruppe User erhält nicht die gewünschten Informationen, ist genervt
			100%						
4. Reuse der Formeln						5%			
4.1	Formeln in anderen Medien reusebar					0	0	Die erstellten Formeln eines Projektes sollen vom Autor auch in anderen Medien wiederverwendet werden können, z.B. bei PowerPoint Präsentationen, in TeX-Papers	z.B. für Paper, in PowerPoint-Präsentationen; Zeitersparnis
			100%						
					Total	100%		99,95%	0

Lösungsvergleich				Lösung 3 *.tex in *.jpg mit texvc umwandeln		
	Ebene 1	Ebene2	Wert	Punkte 1 - 6	Wert	Begründung
1. Installationsaufwand	15%					
1.1 Geringer Installationsaufwand		100%	15,00%	6	90	Es muss keine neue Software installiert werden
		100%				
2. Redaktionelle Tätigkeiten	50%					
2.1 Wenige Schritte zur Durchführung		10%	4,76%	2	9,52	1.Formel in TeX Datei schreiben und Datei mit richtigem Namen/ Verzeichnis abspeichern 2. Aufruf des Konvertierungsporgamms im Internet über URL http://hacks.atrus.org/math_render/ 3.TeX-Code in das Textfeld kopieren und Button drücken 4.Abspeichern der erstellten png-Datei ("expression.png") mit richtigem Namen/ Verzeichnis 5.Im XML-Dokument der Webseite leeres Element einfügen mit dem Attribut ref 6. Bei Attribut ref als Wert Dateipfad angeben
2.2 Automatisierungsgrad hoch (wenig manuell)		14%	7,15%	3	21,45	2.automatisierbar: Aufruf per Batch-Datei oder URL als Lesezeichen von vorne herein abspeichern 6. automatisierbar: Statt Dateipfad Bilddateiname eingeben und XSLT ergänzt bei der Transformation in HTML den Dateipfad
2.3 User kann bekannte Tools verwenden		19%	9,50%	6	57	Ja: Erstellung der Formel mit TeX
2.4 User muss zur Durchführung wenig neue Kenntnisse erwerben		24%	11,90%	5	59,5	Nur der Prozessablauf nötig: richtiges Speicherverzeichnis für LaTeX- und Bilddatei, Vorgehen im XML-Dokument
2.5 Formel in WYSIWYG-Ansicht vom XML-Editor anschaulich		10%	4,76%	6	28,56	PNG anzeigbar (über entsprechende Programmierung); Formel in TeX ebenfalls anzeigbar durch Umwandlung in pdf
2.6 Änderungen werden zentral, an einer Stelle getätigt (Formel; Schriftgröße)		5%	2,38%	6	14,28	Ja: nur in den TeX-Dateien
2.7 Aktualisierungsaufwand nach Änderungen ist gering (Formel; Schriftgröße)		19%	9,50%	4	38	1.Bilder neu generieren: Internetaufruf,Codeeingabe, Abspeichern 2.Bilder auf den Webserver hochstellen. Änderungen an den Webseiten selbst sind nicht durchzuführen
		100%				
3. Webseiten konform	30%					

Lösungsvergleich									Lösung 3 *.tex in *.jpg mit texvc umwandeln		
						Ebene 1	Ebene2	Wert	Punkte 1 - 6	Wert	Begründung
3.1	Schriftgröße der Formel einstellbar						50%	15,00%	1	15	Nein: es kann nur die Formel selbst an den Konverter übergeben werden; eine Übergabe des ganzen LaTeX Dokumentes mit /documentclass und Parameter zur Schriftgröße funktioniert nicht
3.2	Barrierefreiheit/Keine zusätzliche Installation für User						50%	15,00%	5	75	Keine, png-Bilder werden von den aktuellen Webbrowserversionen und auch von den meisten älteren angezeigt
							100%				
4. Reuse der Formeln						5%					
4.1	Formeln in anderen Medien reusebar						100%	5,00%	3	15	Bilder reusable auf Folien in PowerPoint, auf Flyern, Plakaten, die in Corel-Draw und Open.Office.org. Draw erstellt werden, im LaTeX-Format für Papers (wenn gewünscht) Aber: Schrift zu klein für Folien, Plakate
							100%				
					Total	100%		99,95%		318,31	

D. Beispiel für Batch-Dateien

Codeausschnitt der Batch-Datei *check_userinput_call_generate_project.bat*

```
@ECHO OFF
```

```
Rem *****
```

```
Rem Aufgabe der Batch-Datei:
```

```
Rem - Vom User Angaben anfordern und auf Korrektheit überprüfen
```

```
Rem - User-Angaben weiterleiten zur Projektgenerierung an Batch-Datei: "generate_project.bat"
```

```
Rem - User durch Kontrollmeldungen informieren, wie weit die Projektgenerierung vorangeschritten ist
```

```
Rem Voraussetzung: Zugriff auf
```

```
Rem - Batch-Datei: generate_project.bat
```

```
Rem *****
```

```
Rem *****
```

```
Rem Ziel: Befehlsweiterungen für diese Batch-Datei einschalten
```

```
Rem Zweck: DOS-Erweiterungen nutzen
```

```
Rem Schritte/Erläuterung:
```

```
Rem - z.B. in FOR-DO-Schleifen gesetzte Variablen in den Schleifen weiterverwenden
```

```
Rem *****
```

```
setlocal EnableExtensions
```

```
setlocal EnableDelayedExpansion
```

```
Rem *****
```

```
Rem Ziel: Eingabekonsole formatieren
```

```
Rem Zweck: Bessere Lesbarkeit
```

```
Rem Schritte/Erläuterung:
```

```
Rem - Nimm als Titel für die Eingabekonsole "create projects"
```

```
Rem - Nimm als Hintergrundfarbe für die Eingabekonsole weiß (F0)
```

```
Rem *****
```

```
TITLE create projects
```

```
COLOR F0
```

```
Rem *****
```

```
Rem Ziel: Globale Variablen setzen mit dem Wert der Übergabeparameter aus Tool Authentic
```

```
Rem Zweck: Werte an verschiedenen Stellen im Skript wieder verwenden und nur an einer Stelle ändern
```

```
Rem Input: Parameterwerte vom Authentic-Skript "template_script.prj": Formular "generate_new_project"
```

```
Rem Output: Verwendet in den folgenden Abschnitten, um Werte auf Korrektheit hin zu überprüfen
```

```
Rem *****
```

```
SET PROJECT_NAME=%1
```

```
SET PROJECT_TYPE_NR=%2
```

```
SET USER_NAME=%3
```

```
SET PERSON_STATUS_NR=%4
```

```
SET SUPERVISOR_NAME_NR=%5
```

```
SET PROJECT_BEGIN=%6
```

```
SET PROJECT_END=%7
```

```
SET PROJECTS_FOLDER=..\..\projects
```

```
Rem *****
```

```
Rem Ziel: Anführungszeichen in den Werten der globalen Variablen entfernen
```

```
Rem Zweck: Korrekter Programmverlauf
```

```
Rem Input: Werte der globalen Variablen aus vorherigem Programmschritt
```

```
Rem Output: Globale Variablen mit angepassten Werten für die folgenden Programmschritte
```

```
Rem Schritte/Erläuterung:
```

```
Rem - Nimm alten Variablenwert und gebe ihn ab dem 2. Zeichen den Wert aus, entferne aber das letzte Zeichen des Wertes
```

```
Rem *****
```

```
IF NOT "%7"==" " (
```

```
    SET PROJECT_NAME=%PROJECT_NAME:~1,-1%
```

```
    SET PROJECT_TYPE_NR=%PROJECT_TYPE_NR:~1,-1%
```

```
    SET USER_NAME=%USER_NAME:~1,-1%
```

```
    SET PERSON_STATUS_NR=%PERSON_STATUS_NR:~1,-1%
```

```

        SET SUPERVISOR_NAME_NR=%SUPERVISOR_NAME_NR:~1,-1%
        SET PROJECT_BEGIN=%PROJECT_BEGIN:~1,-1%
        SET PROJECT_END=%PROJECT_END:~1,-1%
        GOTO CREATE_DATA
    )

Rem *****
Rem Ziel: Vom User korrekte Eingabe für Projektnamen erhalten
Rem Zweck: Projektnamen weitergeben, um Ordner, Dateien umzubenennen und ihn in XML-Projektdateien als Inhalt für
        Elemente zu verwenden
Rem Output: Projektnamen; Weitergabe an Programmabschnitt "check data"
Rem Schritte/Erläuterung:
Rem - Fordere Benutzer auf Projektnamen einzugeben (/P) und speichere diesen in Variable "PROJECT_NAME"
Rem - Überprüfe, ob Benutzereingabe leer ist (DIR /A:D = nur Dateien auflisten)
Rem - Wenn ja, gib entsprechende Meldung aus und fordere zur nochmaligen Eingabe auf
Rem - Wenn nein, überprüfe, ob der Projektnamen bereits existiert
Rem - Wenn ja, gib entsprechende Meldung aus und fordere zur nochmaligen Eingabe auf
Rem - Wiederhole den bisherigen Vorgang solange, bis der User eine korrekte Eingabe getätigt hat (GOTO
        PROJECT_NAME_INPUT)
Rem - Wenn nein, entferne Leerzeichen vor und hinter dem Projektnamen und ersetze Leerzeichen zwischen den Wörtern
        durch Unterstrich
Rem Zweck: Um korrekte Datei-, Ordnernamen in "generate_project.bat" zu erhalten
Rem - Schreibe angepassten Projektnamen in Datei "project_name.txt"
Rem Zweck: Um Projektnamen von dort in Variable "PROJECT_NAME" speichern zu können
Rem - Lösche die temporäre Datei "project_name.txt"
Rem *****
:PROJECT_NAME_INPUT
SET /P PROJECT_NAME="Please enter the name of your project [%PROJECT_NAME%]:"
ECHO.

IF "%PROJECT_NAME%"==" " (
    ECHO You have forgotten to enter the name of your project
    GOTO PROJECT_NAME_INPUT
) ELSE (
    ECHO %PROJECT_NAME%|SED "s/[ ]*/_g"|SED -n "s/^_*(.[0-9a-zA-Z])_*$\1/p">project_name.txt
    FOR /F "delims=" %%I IN (project_name.txt) DO SET PROJECT_NAME=%%I

    FOR /F "delims=" %%I IN ('DIR /B /A:D %PROJECTS_FOLDER%') DO (
        IF "%%I"=="project_!PROJECT_NAME!" (
            ECHO This project name already exist! Please enter an-other project name!
            GOTO PROJECT_NAME_INPUT
        )
    )
)

)

IF EXIST project_name.txt DEL project_name.txt
Rem *****
Rem Ziel: Vom User korrekte Eingabe für Projekttyp erhalten
Rem Zweck: Projekttyp weitergeben, um abhängig vom Typ, die entsprechenden Ordner, Dateien aus Prototyp-Projekt
        herauszufiltern
Rem Output: Projekttyp(-nummer); Weitergabe an Programmabschnitt "check data"
Rem Schritte/Erläuterung:
Rem - Fordere vom Benutzer(/P) Eingabe des Projekttyps als Nummer und speichere Nummer in Variable
        "%PROJECT_TYPE_NR"
Rem - Überprüfe, ob Eingabe leer ist
Rem - Wenn ja, gib entsprechende Meldung aus und fordere zur nochmaligen Eingabe auf
Rem - Wenn nein, entferne Leerzeichen, falls vorhanden
Rem Zweck: Um in Datei "generate_project.bat" der Numer den Typkürzel korrekt zuweisen zu können
Rem - Speichere Ergebnis in Datei "projectnr.txt"
Rem Zweck: Um Ergebnis von dort in Variable "PROJECT_TYPE_NR" speichern zu können
Rem - Lösche temporäre Datei: "projectnr.txt"

```

```

Rem - Überprüfe, ob die Benutzereingabe einer der vorgegebenen Nummern 1 und 2 entspricht
Rem - Wenn nein, gib entsprechende Meldung aus, lösche Variableninhalt und fordere zur nochmaligen Eingabe auf
Rem - Wiederhole den bisherigen Vorgang solange, bis der User korrekte Eingabe getätigt hat (GOTO
    PROJECT_NAME_INPUT)
Rem Wenn ja, gehe zur nächsten Benutzereingabeforderung (USERNAME_INSERT_CHECK)
Rem *****
:PROJECT_TYPE_INPUT
SET /P PROJECT_TYPE_NR="Please enter the number for the type of your pro-ject (1=Algorithmus, 2=Software_general)
[%PROJECT_TYPE_NR%]:"
ECHO.
IF "%PROJECT_TYPE_NR%"==" " (
    ECHO You have forgotton to enter a number for the type of your project!
    ECHO.
    GOTO PROJECT_TYPE_INPUT
) ELSE (
    ECHO %PROJECT_TYPE_NR%|SED "s/[ ]*/g">projectnr.txt
    FOR /F "delims=" %%J IN (projectnr.txt) DO SET PRO-JECT_TYPE_NR=%%J
    IF EXIST projectnr.txt DEL projectnr.txt
    FOR %%I IN (1 2) DO IF "!PROJECT_TYPE_NR!"=="%%I" GOTO USERNAME_INPUT
    ECHO You haven't used the default numbers 1 or 2 for the type of your project
    ECHO.
    SET PROJECT_TYPE_NR=
    GOTO PROJECT_TYPE_INPUT
)

Rem *****
Rem Ziel: Vom User korrekte Eingabe für Namen erhalten
Rem Zweck: Usernamen weitergeben, um ihn in XML-Projektdateien als Inhalt von Elementen zu verwenden
Rem Output: Username; Weitergabe an Programmabschnitt "check data"
Rem Schritte/Erläuterung:
Rem - Fordere vom Benutzer(/P) Eingabe seines Namens und speichere diesen in Variable "USER_NAME"
Rem - Überprüfe, ob Eingabe leer ist
Rem - Wenn ja, gib entsprechende Meldung aus und fordere zur nochmaligen Eingabe auf
Rem - Wenn nein, überprüfe, ob die Eingabe das obligatorische Komma hat (E-CHO %USER_NAME%|FINDSTR ",")
Rem - Wenn nein (IF ERRORLEVEL 1), gib entsprechende Meldung aus und fordere zur nochmaligen Eingabe auf
Rem - Speichere den Output von FINDSTR in Datei "username.txt"
Rem Zweck: Um Ausgabe in Konsole zu verhindern
Rem - Wiederhole den bisherigen Vorgang solange, bis der User korrekte Eingabe getätigt hat (GOTO
    PROJECT_NAME_INPUT)
Rem - Entferne alle Leerzeichen: vor und hinter dem Komma, vor und hinter dem Namen
Rem - Schreibe angepassten Usernamen in Datei "username.txt"
Rem Zweck: Um Username von dort in Variable "USER_NAME" speichern zu können
Rem - Lösche temporäre Datei "username.txt"
Rem *****
:USERNAME_INPUT
SET /P USER_NAME="Please enter your name (last name,first name, e.g. Holdermann,Fabienne C.)[%USER_NAME%]:"
ECHO.
IF "%USER_NAME%"==" " (
    ECHO You have forgotten to enter your name!
    GOTO USERNAME_INPUT
) ELSE (
    ECHO %USER_NAME%|FINDSTR ", " >username.txt

    IF ERRORLEVEL 1 (
        ECHO You have forgotten the comma between last name and first name: e.g. Holdermann,Fabienne C.
        SET USER_NAME=
        GOTO USERNAME_INPUT
    )
)
ECHO %USER_NAME%|SED "s/[ ]*,[ ]*/,/g"|SED -n "s/^[ ]*(.[0-9a-zA-Z])\ ]*$/1/p">username.txt
FOR /F "delims=" %%I IN (username.txt) DO SET USER_NAME=%%I

```

IF EXIST username.txt DEL username.txt

Rem *****

Rem Ziel: Vom User korrekte Eingabe für seinen Status in der Abteilung erhalten

Rem Zweck: Personenstatus weitergeben, um ihn in XML-Projektdateien als Inhalt von Elementen zu verwenden

Rem Output: Personenstatus-Nummer; Weitergabe an Programmabschnitt "check data" und Voraussetzung für nächsten Programmschritt

Rem Schritte/Erläuterung:

Rem - Fordere vom Benutzer(/P) Eingabe der Personenstatus-Nummer und speichere sie in Variable "PERSON_STATUS_NR"

Rem - Überprüfe, ob Eingabe leer ist

Rem - Wenn ja, gib entsprechende Meldung aus und fordere zur nochmaligen Eingabe auf

Rem - Wenn nein, entferne Leerzeichen, falls vorhanden

Rem Zweck: Um in Datei "generate_project.bat" der Nummer den Personenstatus-Begriff korrekt zuweisen zu können

Rem - Speichere angepasste Personenstatus-Nummer in Datei "statusnr.txt"

Rem Zweck: Um Personenstatus-Nummer von dort in Variable "PERSON_STATUS_NR" speichern zu können

Rem - Lösche temporäre Datei "statusnr.txt"

Rem - Überprüfe, ob die Benutzereingabe einer der vorgegebenen Nummern 1-6 entspricht

Rem - Wenn nein, gib entsprechende Meldung aus, lösche Variableninhalt und fordere zur nochmaligen Eingabe auf

Rem - Wiederhole den bisherigen Vorgang solange, bis der User korrekte Eingabe getätigt hat (GOTO PROJECT_NAME_INPUT)

Rem - Wenn ja, gehe zur nächsten Eingabe :SUPERVISORNAME_INPUT

Rem *****

:PERSONSTATUS_INPUT

SET /P PERSON_STATUS_NR="Please enter the number for your person status (1=student trainee, 2=BA-student, 3=Dipl.-student, 4=MA-student, 5=doctorand, 6=employee)[%PERSON_STATUS_NR%]:"

ECHO.

IF "%PERSON_STATUS_NR%"==" " (

ECHO You have forgotton to enter a number for your person status!

GOTO PERSONSTATUS_INPUT

) ELSE (

ECHO %PERSON_STATUS_NR%|SED "s/[]*/g">statusnr.txt

FOR /F "delims=" %%J IN (statusnr.txt) DO SET PER-SON_STATUS_NR=%%J

IF EXIST statusnr.txt DEL statusnr.txt

FOR /L %%I IN (1,1,6) DO IF "!PERSON_STATUS_NR!"=="%%I" GOTO SUPERVISORNAME_INPUT

ECHO You haven't used the default numbers 1-6 for the type of your project!

ECHO.

SET PERSON_STATUS_NR=

GOTO PERSONSTATUS_INPUT

)

Rem *****

Rem Ziel: Vom User korrekte Eingabe für Betreuernamen erhalten

Rem Zweck: Betreuer Nummer weitergeben, um Betreuernamen in XML-Projektdateien als Inhalt von Elementen zu verwenden

Rem Input: Personenstatus-Nummer aus vorherigem Programmschritt: "PER-SON_STATUS_NR"

Rem Output: Betreuer Nummer; Weitergabe an Programmabschnitt "check data"

Rem Schritte/Erläuterung:

Rem - Fordere vom Benutzer(/P) Eingabe der Betreuer-Nummer und speichere sie in Variable "SUPERVISOR_NAME_NR"

Rem - Überprüfe, ob Eingabe leer ist

Rem - Wenn ja, gib entsprechende Meldung aus und fordere zur nochmaligen Eingabe auf

Rem - Wenn nein, entferne Leerzeichen, falls vorhanden

Rem Zweck: Um in Datei "generate_project.bat" der Nummer den Betreuer Namen korrekt zuweisen zu können

Rem - Speichere angepasste Betreuer-Nummer in Datei "supervisorname.txt"

Rem Zweck: Um Betreuer-Nummer von dort in Variable "SUPERVISOR_NAME_NR" speichern zu können

Rem - Lösche temporäre Datei "supervisorname.txt"

Rem - Überprüfe, ob die Benutzereingabe einer der vorgegebenen Nummern 1-4 entspricht

Rem - Wenn nein, gib entsprechende Meldung aus, lösche Variableninhalt und fordere zur nochmaligen Eingabe auf

Rem - Wiederhole den bisherigen Vorgang solange, bis der User korrekte Eingabe getätigt hat (GOTO PROJECT_NAME_INPUT)

Rem *****

IF %PERSON_STATUS_NR%==6 GOTO PERIOD_INPUT

:SUPERVISORNAME_INPUT

```

SET /P SUPERVISOR_NAME_NR="Please enter the number for your project supervisor (1=Rainer Stotzka, 2=Volker Hartmann,
3=Thomas Jejkal, 4=Michael Sutter, 5=Jochen Haemmerle)[%SUPERVISOR_NAME_NR%].:"
ECHO.
IF "%SUPERVISOR_NAME_NR%"==" " (
    ECHO You have forgotton to enter a number for your project supervisor!
    GOTO SUPERVISORNAME_INPUT
) ELSE (
    ECHO %SUPERVISOR_NAME_NR%|SED "s/[ ]*/g">supervisorname.txt
    FOR /F "delims=" %%J IN (supervisorname.txt) DO SET SUPERVISOR_NAME_NR=%%J
    IF EXIST supervisorname.txt DEL supervisorname.txt
    FOR /L %%I IN (1,1,4) DO IF "!SUPERVISOR_NAME_NR!"=="%I" GOTO PERIOD_INPUT
    ECHO You haven't used the default numbers 1-4 for your project supervisor!
    ECHO.
    SET SUPERVISOR_NAME_NR=
    GOTO SUPERVISORNAME_INPUT
)

Rem *****
Rem Ziel: Vom User korrekte Eingabe für Projektbeginn erhalten
Rem Zweck: Datumsangabe weitergeben, um sie in XML-Projektdateien als Inhalt von Elementen zu verwenden
Rem Output: Projektbeginn; Weitergabe an Programmabschnitt "check data"
Rem Schritte/Erläuterung:
Rem - Fordere vom Benutzer(/P) Eingabe des Projektbeginns und speichere sie in Variable "PROJECT_BEGIN"
Rem - Überprüfe, ob Eingabe leer ist
Rem - Wenn ja, gib entsprechende Meldung aus und fordere zur nochmaligen Eingabe auf
Rem - Wenn nein, überprüfe, ob der obligatorische Bindestrich vorhanden ist
Rem - Wenn nein, gib entsprechende Meldung aus, lösche Variableninhalt und fordere zur nochmaligen Eingabe auf
Rem - Überprüfe, ob Benutzer anstatt Zahlen für Monat den Monatsnamen eingegeben hat
Rem - Wenn ja, gib entsprechende Meldung aus, lösche Variableninhalt und fordere zur nochmaligen Eingabe auf
Rem - Wiederhole den bisherigen Vorgang solange, bis der User korrekte Eingabe getätigt hat (GOTO PROJECTBEGIN_INPUT)
Rem - Wenn nein, entferne Leerzeichen vor und hinter dem Datum
Rem Zweck: Um valide XML-Projektdatei zu erhalten
Rem - Speichere angepassten Projektbeginn in Datei "timeperiod.txt"
Rem Zweck: Um Projektbeginn von dort in Variable "PROJECT_BEGIN" speichern zu können
Rem *****
:PERIOD_INPUT
ECHO Please enter the period specified of your project
ECHO.
:PERIODBEGIN_INPUT
SET /P PROJECT_BEGIN="Begin(yyyy-mm, e.g. 2007-04):"
IF "%PROJECT_BEGIN%"==" " (
    ECHO You have forgotten to enter the date of begin!
    GOTO PERIODBEGIN_INPUT
) ELSE (
    ECHO %PROJECT_BEGIN%|FINDSTR "\-">timeperiod.txt

    IF ERRORLEVEL 1 (
        ECHO You have forgotten the hyphen between year and month: e.g. 2007-04
        SET PROJECT_BEGIN=
        GOTO PERIODBEGIN_INPUT
    )
    ECHO %PROJECT_BEGIN%|FINDSTR "[a-zA-Z]">timeperiod.txt

    IF NOT ERRORLEVEL 1 (
        ECHO Use numbers for month instead of name: e.g. 2007-04 instead of 2007-april!
        SET PROJECT_BEGIN=
        GOTO PERIODBEGIN_INPUT
    )
)

ECHO %PROJECT_BEGIN%|SED -n "s/^[ ]*(.[0-9])[ ]*/p" > timeperiod.txt

```

```

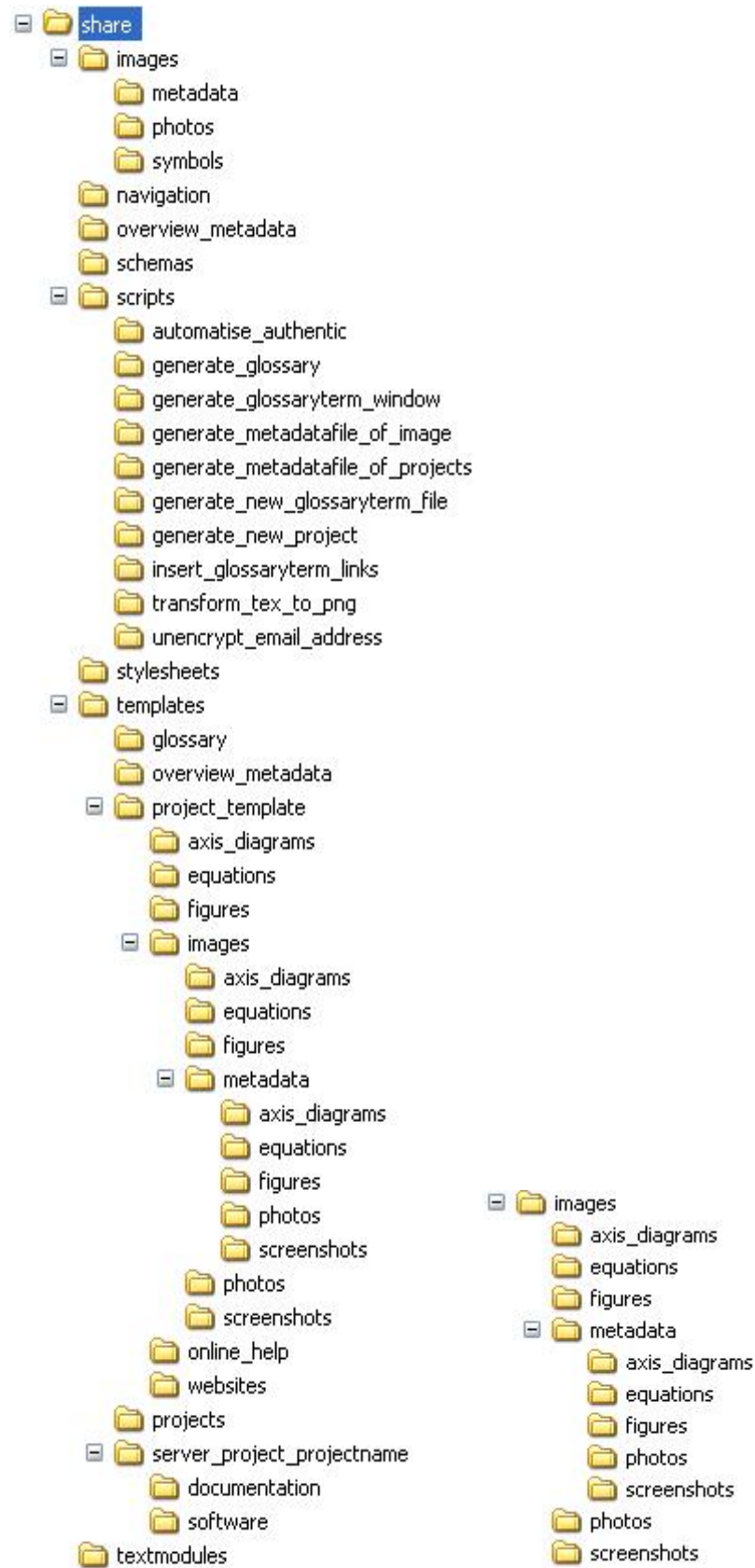
FOR /F "delims=" %%l IN (timeperiod.txt) DO SET PROJECT_BEGIN=%%l
Rem *****
Rem Ziel: Vom User korrekte Eingabe für Projektende erhalten
Rem Zweck: Datumsangabe weitergeben, um sie in XML-Projektdateien als Inhalt von Elementen zu verwenden
Rem Output: Projektende; Weitergabe an Programmabschnitt "check data"
Rem Schritte/Erläuterung:
Rem - Fordere vom Benutzer(/P) Eingabe des Projektendes und speichere sie in Variable "PROJECT_END"
Rem - Überprüfe, ob Eingabe leer ist
Rem - Wenn ja und es handelt sich nicht um Projekt einer Angestellten oder Werkstudenten, gib entsprechende Meldung aus und
      fordere zur nochmaligen Eingabe auf
Rem - Wenn nein, überprüfe, ob der obligatorische Bindestrich vorhanden ist
Rem - Wenn nein, gib entsprechende Meldung aus, lösche Variableninhalt und fordere zur nochmaligen Eingabe auf
Rem - Überprüfe, ob Benutzer anstatt Zahlen für Monat den Monatsnamen eingegeben hat
Rem - Wenn ja, gib entsprechende Meldung aus, lösche Variableninhalt und fordere zur nochmaligen Eingabe auf
Rem - Wiederhole den bisherigen Vorgang solange, bis der User korrekte Eingabe getätigt hat (GOTO PROJECTEND_INPUT)
Rem - Wenn nein, entferne Leerzeichen vor und hinter dem Datum
Rem  Zweck: Um valide XML-Projektdatei zu erhalten
Rem - Speichere angepasste Betreuer-Nummer in Datei "timeperiod.txt",
Rem  Zweck: Um Projektende von dort in Variable "PROJECT_END" speichern zu können
Rem - Lösche temporäre Datei "timeperiod.txt"
Rem *****
:PERIODEND_INPUT
ECHO.
SET /P PROJECT_END="End(yyyy-mm, e.g. 2007-10):"
IF "%PROJECT_END%"==" " (
    IF NOT "%PERSON_STATUS_NR%"=="1" (
        IF NOT "%PERSON_STATUS_NR%"=="6" (
            ECHO You have forgotten to enter the date of begin!
            GOTO PERIODBEGIN_INPUT
        )
    )
) ELSE (
    ECHO %PROJECT_END%|FINDSTR "\-">timeperiod.txt

    IF ERRORLEVEL 1 (
        ECHO You have forgotten the hyphen between year and month: e.g. 2007-10
        SET PROJECT_END=
        GOTO PERIODEND_INPUT
    )
    ECHO %PROJECT_END%|FINDSTR "[a-zA-Z]">timeperiod.txt

    IF NOT ERRORLEVEL 1 (
        ECHO Use numbers for month instead of name: e.g. 2007-10 instead of 2007-october!
        SET PROJECT_END=
        GOTO PERIODEND_INPUT
    )
)
ECHO %PROJECT_END%|SED -n "s/^[\ ]*(\.[0-9])[\ ]*$\n/p" > timeperiod.txt
FOR /F "delims=" %%l IN (timeperiod.txt) DO SET PROJECT_END=%%l
IF EXIST timeperiod.txt DEL timeperiod.txt

```

E. Ordnerstruktur

Abbildung: Struktur der Ordner *share* und *images*

Quelle: Eigene Darstellung

F. Alter und neuer Webauftritt des IPE

Adresse http://www.ipe.fzk.de/projekt/d_index.html

Google Los geht's! Lesezeichen 35 blockiert Rechtschreibprüfung Senden an

Forschungszentrum Karlsruhe
in der Helmholtz-Gemeinschaft

erweiterte Suche Kontakt Sitemap Impressum Intranet

Volltextsuche:

Auf einen Blick Forschungsthemen Organisation Angebote/Service Jobs, Aus-/Fortbildung Akt...

Prozessdatenverarbeitung und

→ IPE → Projekte

Grid Computing

Übersicht Projekte

Alphabetischer Index der Projektnamen

Auger - Observatory for cosmic rays

ELMINA - Künstliche Nase

GRID

Mikrosystemtechnik

MIPAS - Spurengase in der Stratosphäre

Molch - Pipeline -Inspektion

TOSKA - Supraleitende Spulen

USCT

Abgeschlossene Projekte

The idea of GRID computing is to provide worldwide information and computer resources in so far unimaginable quantities. Motivated originally by the huge amounts of data from future high-energy physics experiments, GRID technologies are suited to meet the expected computing requirements of new scientific challenges. The development of GRID technologies is mainly based on two ideas:

1. **Cooperative usage of distributed computer resources** allows the short-term accumulation of huge computer capacities solving novel scientific problems and
2. **Virtualization** allows the exchange of components and resources on the fly and conceals technical details and geographical distribution from the user.

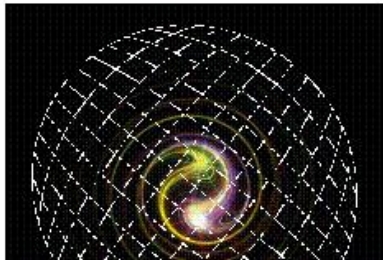


Abbildung: Alter IPE-Auftritt als Basis für Webseitenkonzept

Quelle: http://www.ipe.fzk.de/projekt/d_index.html [Zugriff: 02.07.2007 14:45 MESZ]

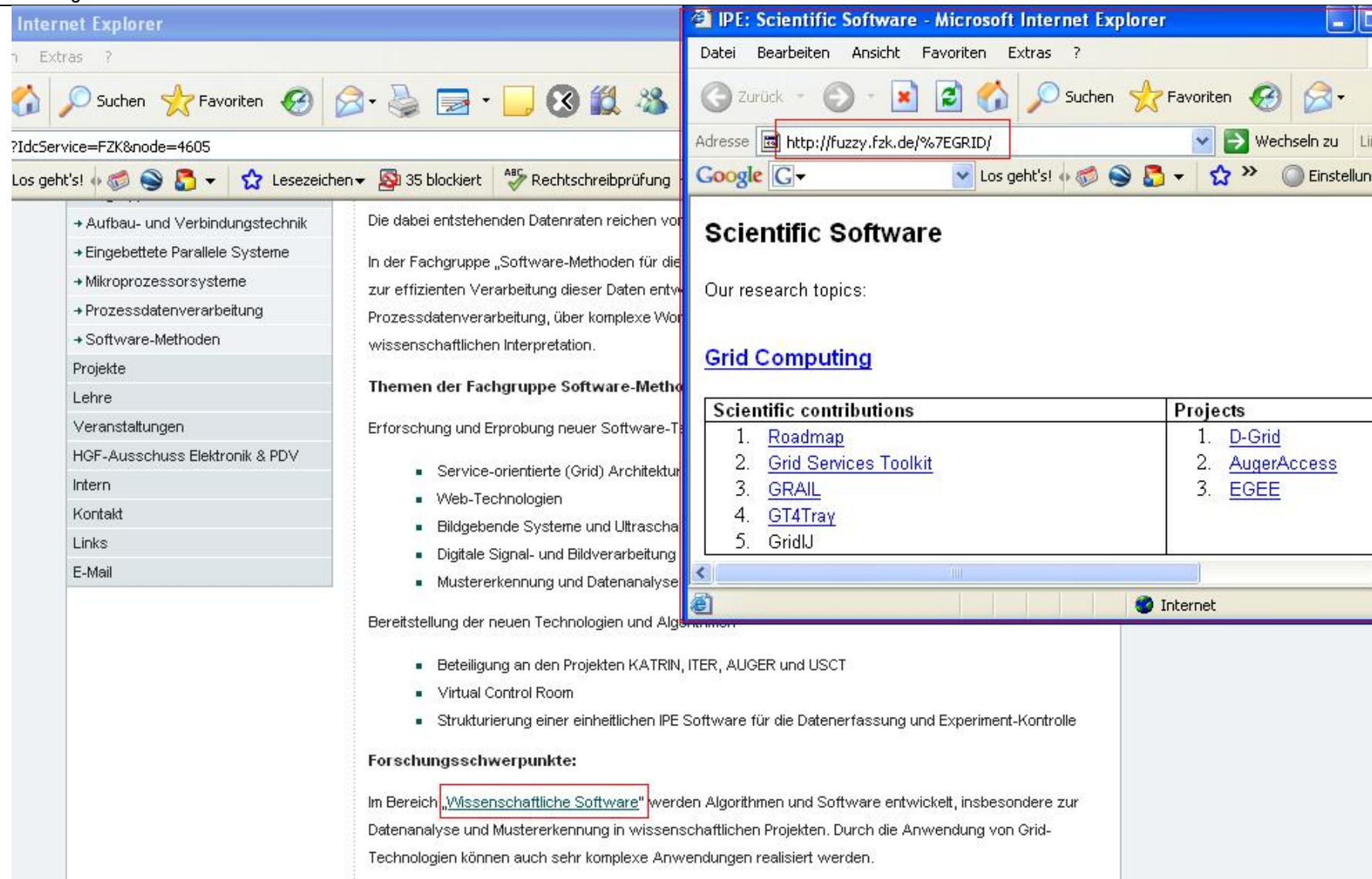


Abbildung: Neuer IPE-Auftritt

Quelle: <http://www.fzk.de/fzk/idcplg?IdcService=FZK&node=4605> [Zugriff: 30.08.2007 11:30 MESZ]

G. Beschreibungsmatrizen zu den Webseitentypen am Beispiel *Projekt-
überblicksseite*

Webseitentyp: Projektüberblicksseite							
Kenndaten							
	Webseitentyp-ID	overview					
	xml/php-Dateinamen-ID	overview_[Projektname]_[Spr.kürzel].xml					
		overview_[Projektname]_[Spr.kürzel].php					
	Beispiel für xml/php-Dateinamen	overview_GT4Tray_en.xml overview_GT4Tray_en.php					
	So sieht die Webseite aus ...						
Lokalisation im Webauftritt							
	Erreichbar von ...	Projektstartseite	Dateinamen-ID: index_[Projektname]_[Spr.kürzel].php				
	Erreichbar sind ...	Glossartermdatei	Vorkommen: optional wenn Webseite Glossarterm				
		Homepage des Auftraggebers	nur wenn Auftraggeber vorhanden				
		Homepage der Partner(abteilungen)	nur wenn Partner vorhanden				
		E-Mail des Ansprechpartners vom Auftraggeber	nur wenn Ansprechpartner vorhanden				
		E-Mail des Ansprechpartners vom Partner(abteilung)	nur wenn Ansprechpartner vorhanden				
		E-Mail der Kontaktperson(en) der Grid-Abteilung	erforderlich				
		E-Mail des Freigebers	nur, wenn er zugestimmt hat				
		Webseite des Bildanbieters pixelio.de	nur, wenn Projektlogo nicht vorhanden ist				
Zeitpunkt Erstellung							
	Wann?	- Kurzfassung: wenn Themenstellung und Zielsetzung definiert sind (Nach Vortrag beim"Mittwochsfrühstück") - Langfassung: wenn stabile Software-version vorhanden ist					
	Von wem?	Vom Projektdurchführenden (Entwickler)					
Funktionsüberblick							
	Funktion der Webseite Webseite gibt Antwort auf...	Überblicks-, Informationsseite, Marketing orientiert:					
		Um was geht es in dem Projekt?					
		Welchen Nutzen hat der User von dem Projekt?					
		Wie lauten die Eckdaten des Projektes?					
	Im Detail heißt das...	Was behandelt das Projekt?	erforderlich in Kurz-/Langfassung				
		Warum wird dieses Projekt durchgeführt?	erforderlich in Langfassung				
		Welche Vorteile ergeben sich für den User?					
		Alternative: Welche Vorteile ergeben sich für den User zu früher? Wie sehen diese Vorteile aus?	erforderlich in Langfassung				

Webseitentyp: Projektüberblicksseite							
		Welche Zielgruppe(n) hat das Projekt?	erforderlich in Langfassung				
		Ist das Projekt als Auftrag von Extern vergeben worden?	optional in Kurz-/Langfassung				
		Besteht eine Zusammenarbeit mit anderen Entwicklergruppen bei dem Projekt?	optional in Kurz-/Langfassung				
		Wer von der Softwaregruppe führt das Projekt durch?	optional in Kurz-/Langfassung				
		Wann wurde das Projekt begonnen?	erforderlich in Kurz-/Langfassung				
		Wann wird es beendet?	erforderlich in Kurz-/Langfassung, wenn es sich um ein studentisches Projekt handelt ansonsten optional				
		Wie weit ist das Projekt vorangeschritten?	erforderlich in Kurz-/Langfassung				
		An wen kann sich der User wenden, wenn er weitere Informationen wünscht?	erforderlich in Kurz-/Langfassung				
	Zu beachten sind...	Will der Freigeber (approver) nicht in der Fußzeile angegeben werden bzw. ohne E-Mail Adresse ist in den Attributen <i>publish</i> und <i>email_publish</i> des Elementes <i>person</i> mit Parent <i>approved</i> der Wert <i>false</i> anzugeben --> Der Freigeber erscheint nicht bzw. seine E-Mail Adresse in der Outputdatei					
		Will der Entwickler nicht genannt werden auf den Projektwebseiten oder in den Metadaten der Webseite, ist im Attribut <i>publish</i> von Element <i>project_collaborator true</i> anzugeben --> Der Entwickler erscheint in keiner Projekt-Outputdatei					
		Bei der Wahl der Schlüsselwörter gilt: Mindestens ein Schlüsselwort aus den Bereichen - Technologien (technology) Bsp.: Web Services - Wissenschaftliche Leistung (scientific) Bsp.: plattformunabhängig, Dynamische Grid-Service-Architektur, neuer Algorithmus					
		Autor muss darauf achten, im Attribut <i>form</i> des Elements <i>content</i> Lang- oder Kurzfassung (<i>elaborated version/abstract</i>) einzustellen					

Webseitentyp: Projektüberblicksseite								
		Für das Projektlogo gilt: - exakte Höhe: 100px - exakte Breite: 100px						
Zielgruppen								
	Nutzertyp	Entwickler, Geldgeber, Interessierte						
	Vorwissen	variiert von Experte bis Anfänger						
Generationsweg								
	Generationsschritte	Die Template-Datei (Ordner <i>templates</i>) enthält die fest eingetragenen Metadaten:						
		Die angelegte Datei enthält die beim Projekt anlegen über Batch und Authentic						
		Autor trägt die geforderte Metadaten ein (<i>overview_of_keyword</i> und <i>publish</i> bei						
		Autor/Bearbeiter überprüft Datei auf Glossarterme mithilfe von Authentic und						
		Autor kennzeichnet das XML-Dokument als <i>created</i> per Buttonklick im Authentic						
		XSL-Stylesheet generiert aus XML-Dokument PHP-Webseite, Textbausteine aus						
		Der Seitenverantwortliche kennzeichnet das XML-Dokument als <i>reviewed</i>						
		Der Seitenverantwortliche kennzeichnet das XML-Dokument als <i>approved</i>						
		XSLT-Stylesheet generiert aus XML-Dokument PHP-Webseite, Textbausteine						
	Der Seitenverantwortliche kennzeichnet je nach Rolle das XML-Dokument als							
	Automatisch eingefügt wird ...	... folgender Inhalt	von wo? wie?					
		Textbausteine für	von XML Datei textmodules.xml:					
		Userangaben: Name und Datum (created, reviewed, approved, published, Metadaten:	per Buttonklick im Editor Authentic					
		Links auf Glossartermdateien	über Batchdatei und Editor Authentic					
		Teil der Keywords bei den Metadaten: siehe Beschreibung metadata_generate.xml	Von Authentic aus über Batch-Dateien					
		Entwicklername auf der Webseite	generiert über metadata_generate.xml					
		Photo des Webseitentyps als Ersatz	wenn Attribut publish=true beim Element <i>project_collaborator</i>					
			Ersatzbild in globaler Variable in overview_project.xml gespeichert					
Aufbau/Struktur								
	Strukturabriss	Navigation	Kurz-/Langfassung					
		Überschrift mit Projektslogan und Bild	Kurz-/Langfassung					
		Einleitung	Kurz-/Langfassung					
		Linkliste mit Links auf die Unterüberschriften	Kurz-/Langfassung					
		Unterüberschrift zur Kurzbeschreibung mit Link zum Seitenanfang	Kurz-/Langfassung					
		Kurzbeschreibung, kann enthalten - Links auf Glossartermdateien - Text hervorhebungen	Kurz-/Langfassung					
		Unterüberschrift zur Motivation mit Link zum Seitenanfang	Langfassung	Alternative:	Unterüberschrift zum Vergleich bisherige Nachteile - Softwarevorteile mit Link zum Seitenanfang			
		Liste mit einzelnen Motivationsgründen als Listenpunkte	Langfassung	Alternative:	3spaltige, mehrzeilige Tabelle: - 1. Spalte: Vergleichskategorie - 2. Spalte: Bisheriger Nachteil/Missstand - 3. Spalte: Vorteil der Software, der den Nachteil/ Missstand behebt			
		Unterüberschrift zu Softwarevorteilen mit Link zum Seitenanfang						
		Einleitung						

Webseitentyp: Projektüberblicksseite					
		Liste mit - Vorteilen als Listenpunkte (Ebene 1) - Vorteile können Unterpunkte besitzen (Listenpunkte Ebene 2)			
		Unterüberschrift zu Zielgruppen der Software mit Link zum Seitenanfang	Langfassung		
		- Zielgruppe - mehrere Zielgruppen als Listenpunkte einer Liste	Langfassung		
		Unterüberschrift zum Entwickler mit Link zum Seitenanfang (optional)	Kurz-/Langfassung		
		- Entwickler (optional) - mehrere Entwickler als Listenpunkte einer Liste (optional)	Kurz-/Langfassung nur wenn Entwickler zugestimmt haben		
		Unterüberschrift zum Auftraggeber mit Link zum Seitenanfang (optional)	Kurz-/Langfassung		
		Tabelle besteht aus 2-5 Zeilen: - Firmenname - Abteilungsname (optional) - Link auf Homepage - Ansprechpartner (optional) -E-Mail Adressen-Link des Ansprechpartner (optional)	Kurz-/Langfassung		
		Unterüberschrift zu den Partnern mit Link zum Seitenanfang (optional)	Kurz-/Langfassung		
		Mehrere Tabellen, je nach Partneranzahl: 1 Tabelle pro Partner besteht aus 2-5 Zeilen: - Firmenname - Abteilungsname (optional) - Link auf Homepage - Ansprechpartner (optional) - E-Mail Adressen-Link des Ansprechpartner (optional) Zwei Tabellen nebeneinander	Kurz-/Langfassung		
		Unterüberschrift zum Projektstatus mit Link zum Seitenanfang	Kurz-/Langfassung		
		Tabelle mit 2-3 Zeilen: - Projektbeginn - Projektende (optional) - Projektstatus	Kurz-/Langfassung		
		Unterüberschrift zu Kontaktpersonen mit Link zum Seitenanfang	Kurz-/Langfassung		

Webseitentyp: Projektübersichtsseite							
		1-2 Tabellen, eine pro Kontaktperson: - Name des Ansprechpartners - Angabe zum Status im Team (optional) - Telefon - Fax - E-Mail Adressen-Link Zwei Tabellen nebeneinander Fußzeile mit ... - Link zum Seitenanfang, - Eckdaten zur Webseite - Hinweis auf Bildanbieter, falls kein Projektbild vorhanden ist	Kurz-/Langfassung				
			Kurz-/Langfassung				
Auszufüllende Elemente vom Autor	Name	Vorfahren	Vorkommen	Datentyp	Funktion	Attribute	
	audience	Child von content	erforderlich bei Langfassung	string	eine Zielgruppe der Software	-	
	benefit_data	Descendant von overview_of_benefits	erforderlich bei Langfassung, wenn overview_of_motivation vorhanden ist	string (mixed content)	ein Vorteil der Software (Listenpunkt)	-	
	category_of_comparison	Descendant von overview_of_comparisons	erforderlich bei Langfassung, wenn overview_of_comparisons vorhanden	string (mixed content)	Vergleichskategorie: für Nachteil früher - Softwarevorteil jetzt	-	
	department	Child von company Descendant von orderer	optional bei Kurz- und Langfassung	string	Abteilung des Auftraggebers	-	
		Child von company Descendant von - overview_of_partners - partner	optional bei Kurz- und Langfassung	string	Abteilung eines Projektpartners	-	
	email	Descendant von orderer	erforderlich bei Kurz- und Langfassung, wenn contact vorhanden	String mit Einschränkung (@, .de)	E-Mail Adresse des Ansprechpartners des Auftraggebers	-	
		Descendant von - overview_of_partners - partner	erforderlich bei Kurz- und Langfassung, wenn contact vorhanden	String mit Einschränkung (@, .de)	E-Mail Adresse des Ansprechpartners eines Projektpartners	-	
		Descendant von overview_of_contacts	erforderlich bei Kurz- und Langfassung	String mit Einschränkung (@, .de)	E-Mail adresse einer Ansprechperson für das Projekt in der Grid-Abteilung	-	
	end	Child von project_status	optional bei Kurz- und Langfassung	gYeareMonth	Projektende (z.B. 2007-10)	-	
	fax	Descendant von overview_of_contacts	erforderlich bei Kurz- und Langfassung	String mit Einschränkung	Fax-Nummer	-	
	first_name	Descendant von contact und orderer	erforderlich bei Kurz- und Langfassung, wenn contact vorhanden bei orderer	String	Vorname des Ansprechpartners des Auftraggebers	-	

Webseitentyp: Projektübersichtsseite							
			Descendant von - overview_of_partners - contact - partner	erforderlich bei Kurz- und Langfassung, wenn overview_of_partners vorhanden	String	Vorname des Ansprechpartners eines Partners	-
			Descendant - overview_of_contacts - contact	erforderlich bei Kurz- und Langfassung	String	Vorname einer Ansprechperson für das Projekt in der Grid-Abteilung	-
		last_name	Descendant von - contact - orderer	erforderlich bei Kurz- und Langfassung, wenn <i>contact</i> vorhanden	String	Nachname des Ansprechpartners des Auftraggebers	-
			Descendant von - contact - overview_of_partners - partner	erforderlich bei Kurz- und Langfassung, wenn <i>contact</i> vorhanden	String	Nachname des Ansprechpartners eines Partners	-
			Descendant von - overview_of_contacts - contact	erforderlich bei Kurz- und Langfassung	String	Nachname einer Ansprechperson für das Projekt in der Grid-Abteilung	-
		logo	Child von topic	optional bei Kurz- und Langfassung	image (leer)	Projektlogo, das auf allen Projektwebseiten neben der Überschrift erscheint	file_path
		markup	Child von short_description Child von motivation Child von benefit_data Child von sub_benefit Child von category_of_comparison Child von past_disadvantage Child von present_advantage	optional bei Kurz- und Langfassung	String	je nach Auswahl des Attributwertes von type hat markup unterschiedliche Funktion: es hebt den Text hervor, - durch Fettdruck - durch Kursivschrift - es setzt zwischen Wörtern ein geschütztes Leerzeichen, damit der Text dort nicht umbricht, z.B. bei Eigennamen: Globus Toolkit	type
		motivation	Child von content	erforderlich bei Langfassung, wenn overview_of_motivation vorhanden ist	string (mixed content)	ein Grund, warum das Projekt durchgeführt wird (Listenpunkt)	-
		name	Child von company Descendant von orderer	erforderlich bei Kurz- und Langfassung, wenn Auftraggeber vorhanden	string	Firmenname des Auftraggebers	-
			Child von company Descendant von - overview_of_partners - partner	erforderlich bei Kurz- und Langfassung, wenn overview_of_partners vorhanden	string	Firmenname eines Partners	

Webseitentyp: Projektüberblicksseite							
		overview_of_keywords	Child von document_data Descendant von metadata	erforderlich	string	Schlüsselwörter für die Webseite, die durch Komma voneinander getrennt werden Mindestens ein Schlüsselwort aus den Bereichen - Technologien (technology) Bsp.: Web Services - Wissenschaftliche Leistung (scientific) Bsp.: plattformunabhängig, Dynamische Grid-Service-Architektur, neuer Algorithmus - identifizieren eine Seite, erleichtern ihr Auffinden - werden als Metadaten für die Webseite und für Kommentare im SVN verwendet	
		past_disadvantage	Descendant von overview_of_comparisons	erforderlich bei Langfassung, wenn overview_of_comparisons vorhanden	string (mixed content)	Nachteil, der bisher bestand - bevor die Software entwickelt wurde	-
		phone	Descendant von overview_of_contacts	erforderlich bei Kurz- und Langfassung	String mit Einschränkung	Telefonnummer einer Ansprechperson für das Projekt in der Grid-Abteilung	-
		present_advantage	Descendant von overview_of_comparisons	erforderlich bei Langfassung, wenn overview_of_comparisons vorhanden	string (mixed content)	Vorteil der Software im Gegensatz zu vorher	-
		progress	Child von project_status	erforderlich bei Kurz- und Langfassung	Enumeration: - Planned - In progress - Finished	Projektstatus	-
		short_description	Child von content	erforderlich bei Kurz- und Langfassung	string (mixed content)	Kurze Beschreibung	-
		slogan	Child von topic	erforderlich bei Kurz- und Langfassung	string	- Projektslogan drückt den Hauptzweck des Projektes pfiffig aus - erscheint auf allen Projektwebseiten und der Überblicksseite der Projekte unter der Überschrift	-

Webseitentyp: Projektübersichtsseite							
		start	Child von project_status	erforderlich bei Kurz- und Langfassung	gYearMonth	Projektbeginn: 2007-10	-
		sub_benefit	Descendant von overview_of_benefits	erforderlich bei Kurz- und Langfassung, wenn overview_of_subbenefits vorhanden	string (mixed content)	Unterpunkt eines Vorteils (benefit_data), dient der weiteren Untergliederung	-
		website_address	Child von company Descendant von orderer	erforderlich bei Kurz- und Langfassung, wenn Auftraggeber vorhanden	link (leer)	Link auf die Homepage des Auftraggebers	href link_text link_type
			Child von company Descendant von - overview_of_partners - partner	erforderlich bei Kurz- und Langfassung, wenn overview_of_partners vorhanden	link (leer)	Link auf die Homepage eines Partners	href link_text link_type
	Auszufüllende Attribute vom Autor	Name	Vorkommen in Elementen	Vorkommen	Datentyp	Werte	Funktion
	file_path	logo	erforderlich	string	-	Dateipfad des Projektlogos wird auf allen Projektseiten neben der Überschrift angezeigt	
	form	content	erforderlich bei Kurz- und Langfassung	Enumeration	- abstract - elaborated_version	dient dazu, die Seite über XSL korrekt auszugeben	
	position	contact (Child von overview_of_contacts)	optional	Enumeration	- Group leader - Deputy group leader	Dient der Spezifizierung des Ansprechpartners	
	publish	project_collaborator	erforderlich	boolean	true, false	true= Entwickler wird namentlich auf - der Projektübersichtsseite - der Downloadseite genannt - wird in den Metadaten der Web-seite als keyword aufgeführt	
	type	markup	nur, wenn Element markup verwendet wird	Enumeration	- hard space - user input - system output - window - menu item - option	- geschütztes Leerzeichen - Kursivschrift - Kursivschrift - Fettdruck - Fettdruck - Fettdruck	
Auszufüllende Attribute vom Approver	Name	Vorkommen in Elementen	Vorkommen	Datentyp	Werte	Funktion	
	publish	person (Parent: approved)	erforderlich	boolean	true, false	regelt Veröffentlichung des Approver	
	email_publish	person (Parent: approved)	erforderlich	boolean	true, false	regelt Veröffentlichung seiner E-Mail Adresse	
	email	person (Parent: approved)	optional	string	E-Mail Adresse	E-Mail Adresse steht zur Verfügung, wenn email_publish mit true beantwortet	
Schemas	Zusammenspiel von...	Datei					
	Webseiten-Schema	overview_project.xsl					

Webseitentyp: Projektüberblicksseite							
		Referenzierte Schemas	share_metadata.xsd share.xsd				
XSL-Stylesheets	Zusammenspiel von...	Datei	Datei-ID				
	Webseiten-Stylesheet	overview_project.xsl	overview_project				
	Referenzierte Stylesheets	contacts_generate.xsl	contacts_generate				
		email_link.xsl	email_link				
		footer_generate.xsl	footer				
		glossaryterm_topic_link.xsl	glossaryterm_topic_link				
		graphic_generate.xsl	graphic				
		headlines_generate.xsl	headline_subheadline				
		metadata_generate.xsl	metadata				
		navigation_insert.xsl	navigation				
		section_link.xsl	section_link				
		shared_textmodule_variables.xsl	shared_textmodules				
		table_generate.xsl	table_generate				
		text_markup.xsl	text_markup				
		top_link.xsl	top_link				
	Referenzierte JavaScript-Scripts	unencryption_email.js	-				
		generate_glossaryterm_window.js	-				
Links	Linkarten	Link-ID					
	Intra-Webseiten-Link	top_link					
		section_link					
	Inter-Webseiten-Link	glossaryterm_topic_link					
	Extra-Webseiten-Link	email_link					
		member_www_link					
		footer_member_www_link					
Textbausteine	Gespeichert in ...	Inhalt					Gespeichert in Datei ...
Globale Variablen	introduction_abstract	Im Folgenden erfahren Sie, was das Projekt behandelt und erhalten grundlegende Eckdaten zum Projekt:					overview_project.xsl
	introduction_elaborated_version	Im Folgenden erfahren Sie alles Wichtige zum Thema, den Vorteilen und Zielgruppen des Projektes sowie seinen					overview_project.xsl
	introduction_overview_of_audiences	Interessant ist die Software für...					overview_project.xsl
	introduction_overview_of_benefits	Die Software bietet dem User folgende Vorteile:					overview_project.xsl
	introduction_overview_of_comparisons	Die Software bietet im Vergleich zu vorher folgende Verbesserungen:					overview_project.xsl
	subheadline_developer	Entwickler					overview_project.xsl
	tablerow_person	Ansprechpartner:					overview_project.xsl
	tablerow_website_address	Webseite:					shared_textmodule_variables.xsl
		Website:					
	tooltip_member_www_link	Webseite von Website of					shared_textmodule_variables.xsl
Direkt aus Textmodul-Datei ausgelesen	Tabellenspaltenbeschriftung von Tabelle overview_of_comparisons	Kategorie Category				-	
	Tabellenzeilenbeschriftung von Tabelle Projektstatus	Beginn: Start:				-	
	Unterüberschrift für Motivation	Motivation				-	
	Unterüberschrift für Auftraggeber	Auftraggeber				-	
	Unterüberschrift für Zielgruppen	Zielgruppen				-	
	Unterüberschrift für Vorteile	Vorteile				-	

Webseitentyp: Projektüberblicksseite				
		Unterüberschrift für Vorteilsvergleich mit früher	Vorteile der Software Advantages of the software	-
		Unterüberschrift für Kontakt	Kontakt	-
		Unterüberschrift für Projektpartner	Projektpartner	-
		Unterüberschrift für Projektstand	Projektstand	-
		Unterüberschrift für Kurzbeschreibung	Kurzbeschreibung	-
	Globale Parameter	Name	Inhalt	übergeben von ...
		-	-	übergeben an ...
	Lokale Parameter	Name	Inhalt	übergeben von ...
		photopath	relativer Dateipfad des Webseitentyp-Bildes von overview_project Gespeichert in Variable photo_overview_project_website	übergeben an Funktion ...
			Template für Root in overview_project.xml	logo_check in XSL-Stylesheet graphic_generate.xml
	Globale Variablen	Name	Inhalt	Gespeichert in Datei ...
		css_path	relativer Dateipfad zum CSS Stylesheet	overview_project.xml
		email_js_path	relativer Dateipfad zur JavaScript Datei für die Entschlüsselung der E-Mail	overview_project.xml
		file_path_part	relativer Dateipfad zum Ordner share und glossary	overview_project.xml
		glossaryterm_js_path	relativer Dateipfad zur JavaScript Datei für das Generieren des Pop-up Fensters für Link auf Glossartermdatei	overview_project.xml
		language	Sprache der Webseite (de/en)	overview_project.xml
		overview_path	relativer Dateipfad zur Überblicksdatei des Projektes: overview_[Projektname]_[Spr.kürzel].xml	overview_project.xml
		photo_overview_project_website	relativer Dateipfad zu Ersatzbild für Überschrift	overview_project.xml
		softwarename	Projektname	overview_project.xml
		symbol_bullet_point	relativer Dateipfad zu Auflistungssymbol	overview_project.xml
		textmodule_path	relativer Dateipfad zur Textmodul-Datei	shared_textmodule_variables.xml
		Textbaustein-Variablen	siehe Textbausteine	siehe Textbausteine
		website_id	overview_project	overview_project.xml
	Lokale Variablen	Name	Inhalt	Gespeichert in Funktion/Template...
		node_content	Elementname des aktuellen Knotens ohne Namensraum	Template overview_of_comparisons Template_end Template_progress Template_start
		publish_developer	Anzahl der Entwickler, die der Veröffentlichung ihres Namens zugestimmt haben	Funktion check_project_collaborator
	Funktionen	Name	Aufgabe	aufgerufen von...
		logo_check	überprüft, ob Projektlogo vorhanden ist fügt Logo oder Ersatzbild ein: in XSL-	Template für Root in overview_project.xml
		top_link_anker_generate	erzeugt Marke für Top-Links: in XSL-Stylesheet top_link.xml	Template für Root in overview_project.xml
		navigation_insert	fügt die Navigation ein: in XSL-Stylesheet navigation_insert.xml	Template für Root in overview_project.xml
		metadata_generate	fügt Metadatenabschnitt ein: in XSL-Stylesheet metadata_generate.xml	Template für Root in overview_project.xml
		headline_generate	fügt Überschrift ein: in XSL-Stylesheet headlines_generate.xml	Template für Root in overview_project.xml
		footer_generate	fügt Fußzeile ein: in XSL-Stylesheet footer_generate.xml	Template für Root in overview_project.xml
		section_link_generate	fügt Linkliste auf Unterüberschriften ein: in XSL-Stylesheet section_link.xml	Template für Root in overview_project.xml
		subheadline_generate	fügt Unterüberschrift ein: in XSL-Stylesheet headlines_generate.xml	In overview_project.xml Template von
		check_project_collaborator	generiert Abschnitt zu Entwicklern, sobald einer der Entwickler die Veröffentlichung	In overview_project.xml
		tablerow_generate	fügt Tabellenzeilen ein: in XSL-Stylesheet table_generate.xml	Template für overview_of_comparisons in overview_project.xml
		top_link	fügt Link zum Seitenanfang ein mit Tooltip	Funktion check_project_collaborator in overview_project.xml
Layout				

Webseitentyp: Projektüberblicksseite						
	Stylesheet-Name	grid_web.css				
	Verwendete Klassen	dotline				
		listsquare_level1				
		listsquare_level2				
		projectlogo_position				
		subheadline				
		tablehead_middle3				
		textsection				
		top_link_column				
		website_position				
		website_table_general				
		website_table_general_column0				
		website_table_general_column1				
		website_table_general_column2				
		website_table_headline				
		website_table_subheadline				
		website_table_two_left				
		website_table_two_right				
	Verwendete ID	headline				
		slogan				
	Autentic-Stylesheet	overview_project.sps				
Verwendete Bilder		Bild(-funktion)	Dateiname	Name der globalen Variable	Dateipfad	
	Headline-Bilder	Ersatzbild für Projektlogo in der Webseitenüberschrift	overview_glasses.jpg	photo_overview_project_website	share/images/photos	
Zu beachten ist für den		Folgendes ...				
	-	-				

H. Grafische Darstellung der Linktypbeziehungen

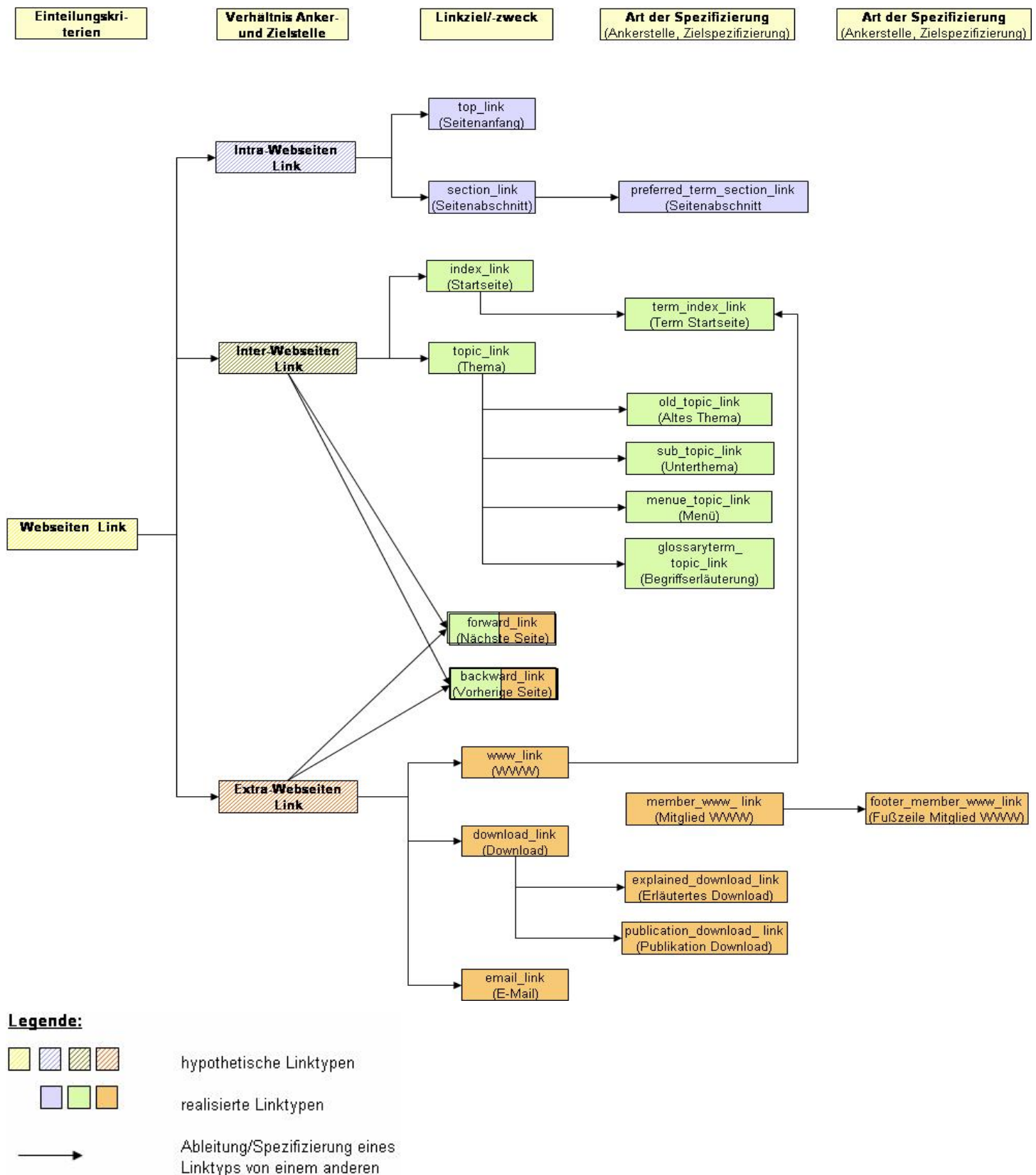
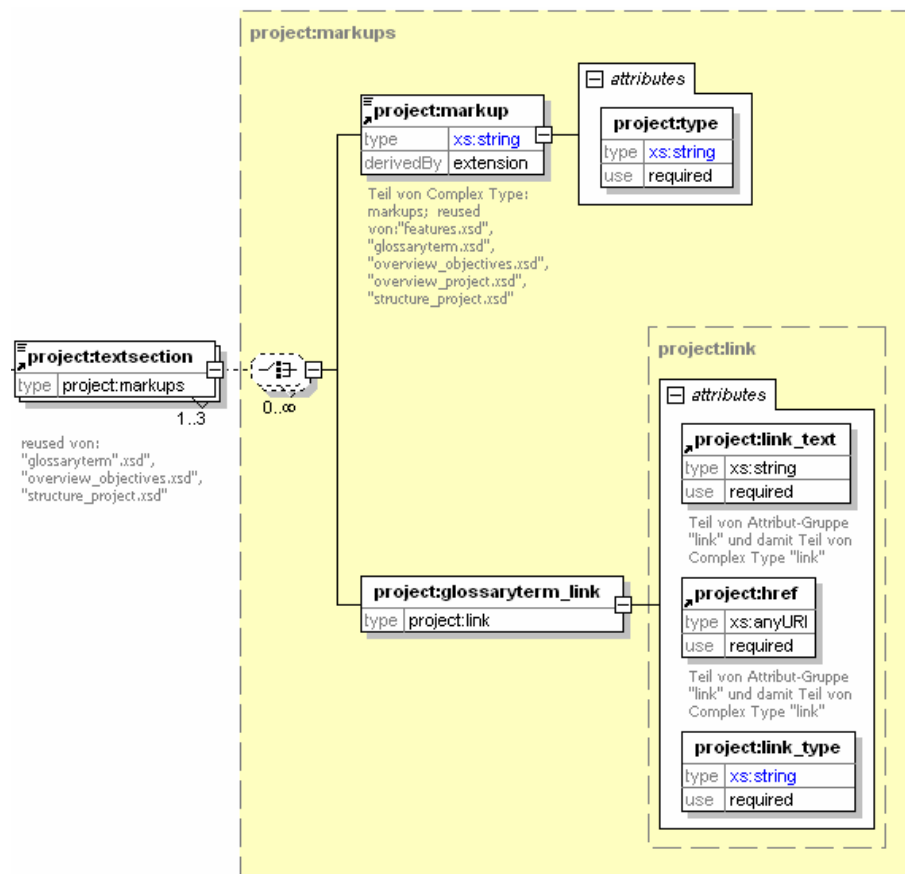


Abbildung: Beziehungen der Linktypen zueinander

Quelle: Eigene Darstellung

I. Beispiele für Complex Types

Abbildung: Complex Type *markups*Quelle: Ausschnitt aus *glossaryterm.xsd*

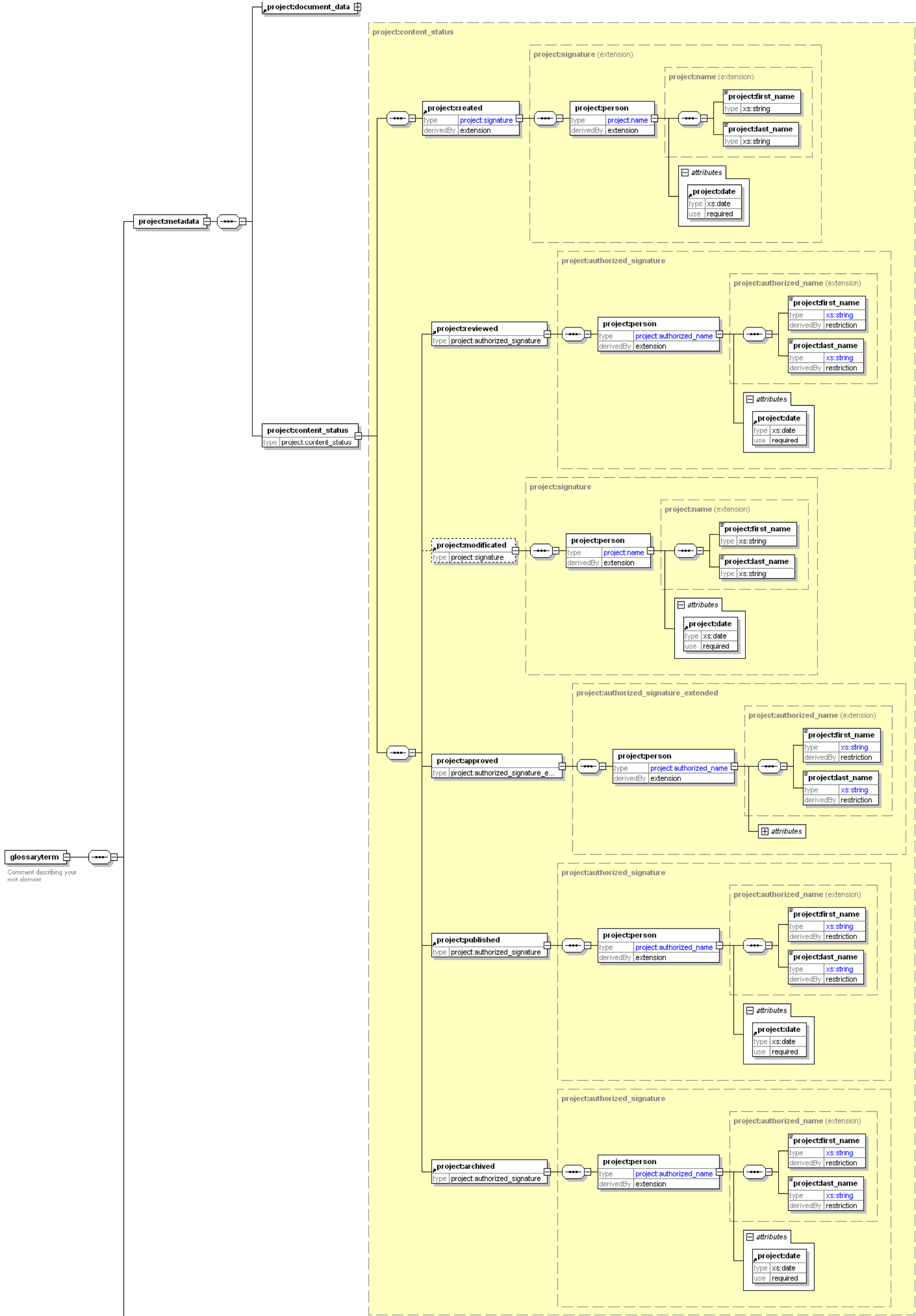


Abbildung: Complex Types für Metadaten
Quelle: Ausschnitt aus *glossaryterm.xsd*

J. Beschreibungsmatrizen der modularen Stylesheets an den Beispielen
metadaten_generate.xsl und *email_link.xsl*

Webseitenelement: E-Mail Link						
Kenndaten						
	Stylesheet-Name	email_link.xsl				
	Stylesheet-ID	email_link				
	Stylesheet-Funktion	- Erzeugt E-Mail Link, der ein Outlook-Fenster mit der E-Mail Adresse öffnet - verschlüsselt E-Mail Adresse				
	So sieht der E-Mail-Link aus ...	E-Mail:  Kurt.Kost@ipe.fzk.de				
		Briefsymbol				
		E-Mail Adresse (At- und Punktzeichen sind Bilder)				
Webseitenelement						
	E-Mail-Link vorhanden in Datei ...	contact_[Spr.kürzel].php download_[Projektname]_[Spr.kürzel].php features_[Projektname]_[Spr.kürzel].php news_[Spr.kürzel].php objectives_[Spr.kürzel].php overview_[Projektname]_[Spr.kürzel].php structure_[Projektname]_[Spr.kürzel].php structure_[Komponentenname]_[Spr.kürzel].php technical_data_[Projektname]_[Spr.kürzel].php	erforderlich optional optional optional optional erforderlich optional optional optional			
	Von E-Mail Link aus sind erreichbar ...	Outlook-Fenster mit E-Mail-Adresse				
Funktionsüberblick des Webseitenelements						
	Funktion des E-Mail Links	Kontaktpersonen und Verantwortliche des Inhalts sollen für den User schnell erreichbar sein				
	E-Mail Link gibt im Detail Antwort auf ...	-				
Aufbau/Struktur des Stylesheets						
	Stylesheet wird referenziert von ...	... folgenden Dateien				
		actual_projects.xsl contact.xsl download_project.xsl features_project.xsl news.xsl objectives.xsl old_projects.xsl overview_project.xsl structure_project_structure_component.xsl technical_data_project.xsl				
	Auf Stylesheet wird zuriffen von ...	... folgenden Dateien				

Webseitenelement: E-Mail Link						
		contacts_generate.xsl footer_generate.xsl				
Stylesheet greift zu auf ...	... folgende Dateien	Inhalt				
	textmodules.xml	Textknoten				
	shared_textmodule_variables.xsl	Variablen für ausgelesene Textknoten				
Stylesheet enthält ...	... folgende Elemente	Vorkommen				
	Textbausteine	x				
	Globale Variablen	x				
	Lokale Variablen	x				
	Funktionen	x				
	Globale Parameter	-				
	Lokale Parameter	x				
	Links	x				
	JavaScript Abschnitte	-				
	Bilder	x				
Textbausteine gespeichert in ...	... folgenden Variablen	Inhalt				Gespeichert in Datei ...
	tooltip_email_link	E-Mail E-mail				email_link.xsl
Globale Variablen	Name	Inhalt				Gespeichert in Datei ...
	at	relativer Bilddateipfad				email_link.xsl
	file_path_part	relativer Dateipfad zum Ordner <i>share</i> und <i>glossary</i>				XSL-Stylesheet, das <i>email_link.xsl</i> referenziert
	language	Sprache der Webseite (de/en)				XSL-Stylesheet, das <i>email_link.xsl</i> referenziert
	point	relativer Bilddateipfad				email_link.xsl
	symbol_email_link	relativer Bilddateipfad				email_link.xsl
	textmodule_path	relativer Dateipfad zur Textmodul-Datei				shared_textmodule_variables.xsl
	Textbaustein-Variable	siehe Textbausteine				siehe Textbausteine
Lokale Variablen	Name	Inhalt				Gültig für Funktion ...
	email_front_of_dot	Vorname				email_generate
	email_front_of_at	Vor- und Nachname				email_generate
	email_behind_dot	Nachname				email_generate
	email_behind_at	ipe.fzk.de				email_generate
	email_front_of_de	ipe.fzk				email_generate
	email_encrypted	verschlüsselte E-Mail-Adresse				email_generate
Funktionen	Name	Aufgabe			aufgerufen von...	

Webseitenelement: E-Mail Link						
		email_generate	- erzeugt Linksymbol mit Tooltipp (Brief) - splittet E-Mail-Adresse in mehrere Textbausteine auf - verschlüsselt E-Mail-Adresse - erzeugt Link: - mit Linktext, der anstelle von @ und Punkten zwischen Vor-, Nachname und vor "de" Bilder hat - mit Tooltipp - mit Aufruf der JavaScript Funktion <i>MailTo</i> im Attribut <i>href</i> und Übergabe der verschlüsselten E-Mail-Adresse, um diese zu entschlüsseln		- Funktion <i>footer_generate</i> (in XSL-Stylesheet <i>contacts_generate.xml</i>) - Template <i>email</i> (in XSL-Stylesheet <i>contacts_generate.xml</i>)	
	Globale Parameter	Name	Inhalt	übergeben von ...	übergeben an ...	
		-	-	-	-	
	Lokale Parameter	Name	Inhalt	übergeben von ...	übergeben an Funktion ...	
		Email	E-Mail Adresse	- Funktion <i>footer_generate</i> (in <i>contacts_generate.xml</i>) - Template <i>email</i> (in <i>contacts_generate.xml</i>)	email_generate	
		style	website_position_footer	- Funktion <i>footer_generate</i> (in <i>contacts_generate.xml</i>) - Template <i>email</i> (in <i>contacts_generate.xml</i>)	email_generate	
			leer	- Funktion <i>footer_generate</i> (in <i>contacts_generate.xml</i>) - Template <i>email</i> (in <i>contacts_generate.xml</i>)	email_generate	
	Links	Linkarten	Link-ID			
		Extra-Webseiten-Link	email_link			
	JavaScript Abschnitte	Name	Funktion			
		-	-			
Layout						
	Stylesheet-Name	grid_web.css				
	Verwendete Klassen	footer_image				
		website_position_footer				
	Verwendete ID	-				
	Verwendete Bilder	Bild(-funktion)	Dateiname	Name der globalen Variable	Dateipfad	
	Linksymbole	At-Zeichen	email_at.gif	at	share/images/symbols	
		Punkt	email_dot.gif	point	share/images/symbols	
		Briefsymbol	email_link_letter.gif	symbol_email_link	share/images/symbols	

Webseitenelement: Metadatenabschnitt						
Kenndaten						
	Stylesheet-Name	metadata_generate.xsl				
	Stylesheet-ID	metadata				
	Stylesheet-Funktion	erzeugt im Kopf der Output-Datei den HTML Metadatenabschnitt				
	So sieht der Metadatenabschnitt aus ...	<pre> <head> <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1"> <meta name="keywords" content="Grid, software, softwaregroup,science, development,GridIJ, Webservice, Bildservice, <meta name="date" content="1111-11-11"> <meta name="DC.title" content="Download of GridIJ"> <meta name="DC.date" content="1111-11-11" scheme="DCTERMS.W3CDTF"> <meta name="DC.language" content="en" scheme="DCTERMS.RFC3066"> <meta name="DC.type" content="Text" scheme="DCTERMS.DCMIType"> <meta name="DC.format" content="text/html" scheme="DCTERMS.IMT"> <meta name="DC.coverage" content="Forschungszentrum Karlsruhe"> <meta name="DC.rights" content="The copyright and all other rights lie with the section Software Methods of the In <title>Download of GridIJ</title> </head> </pre>				
	Meta-Tags					
	Metadaten-Angaben nach Dublin Core					
Webseitenelement						
	Metadatenabschnitt vorhanden in Datei ...	index_[Spr.kürzel].php objectives_[Spr.kürzel].php actual_projects_[Spr.kürzel].php old_projects_[Spr.kürzel].php actual_publications_[Spr.kürzel].php old_publications_[Spr.kürzel].php glossary_[Spr.kürzel].php contact_[Spr.kürzel].php news_[Spr.kürzel].php index_[Projektname]_[Spr.kürzel].php overview_[Projektname]_[Spr.kürzel].php features_[Projektname]_[Spr.kürzel].php structure_[Projektname]_[Spr.kürzel].php structure_[Komponentenname]_[Spr.kürzel].php technical_data_[Projektname]_[Spr.kürzel].php download_[Projektname]_[Spr.kürzel].php overview_metadata_projects_[Spr.kürzel].php	erforderlich erforderlich erforderlich erforderlich erforderlich erforderlich erforderlich erforderlich erforderlich erforderlich erforderlich erforderlich erforderlich erforderlich			
	Von Metadatenabschnitt aus sind erreichbar ...	-				
Funktionsüberblick des Webseitenelements						

Webseitenelement: Metadatenabschnitt						
	Funktion des Metadatenabschnitt	- enthält Suchbegriffe für Suchmaschinen, um eine hohe Trefferquote für jede Seite zu erzielen - Informationen für Web-Browser - identifiziert eine Seite				
	Metadatenabschnitt gibt im Detail Antwort auf ...	Was für Suchbegriffe/Schlüsselwörter gibt es?				
		Was wurde die Webseite freigegeben?				
		Wie heißt die Datei?				
		In welcher Sprache ist der Inhalt verfasst?				
		Welchen Inhaltstyp hat die Webseite?				
		Welches Datenformat hat die Datei?				
		Welchen geographischen Bezugspunkt hat die Webseite?				
		Welches Copyright gilt für den Dateiinhalt?				
Aufbau/Struktur des Stylesheets						
	Stylesheet wird referenziert von ...	... folgenden Dateien				
		index_index_project.xml objectives.xml projects.xml glossary.xml contact.xml news.xml overview_project.xml features_project.xml structure_project_structure_component.xml technical_data_project.xml download_project.xml overview_metadata_projects.xml				
	Auf Stylesheet wird zugegriffen von ...	... folgenden Dateien				
		footer_generate.xml headlines_generate.xml				
	Stylesheet greift zu auf ...	... folgende Dateien	Inhalt			
		textmodules.xml shared_textmodule_variables.xml	Textknoten Variablen für ausgelesene Textknoten			
	Stylesheet enthält ...	... folgende Elemente	Vorkommen			
		Textbausteine	x			
		Globale Variablen	x			
		Lokale Variablen	x			
		Funktionen	x			
		Globale Parameter	-			
		Lokale Parameter	-			

Webseitenelement: Metadatenabschnitt					
		Links	-		
		JavaScript Abschnitte	-		
		Bilder	-		
Textbausteine gespeichert in ...	... folgenden Variablen	Inhalt	Gespeichert in Datei ...		
	copyright_text_metadata	Copyright und alle weiteren Rechte liegen bei der Fachgruppe Software-Methoden des Instituts für Prozessdatenverarbeitung und Elektronik des Forschungszentrums Karlsruhe; siehe auch Impressum The copyright and all other rights lie with the section Software Methods of the Institute for Data Processing and Electronics of the Forschungszentrum Karlsruhe; quod vide masthead	metadata_generate.xsl		
	keywords_fixed	Grid, Software, Softwaregruppe, Wissenschaft, Entwicklung, Grid, software, group of software methods, Grid computer scientists, science, development,	metadata_generate.xsl		
	metadata_coverage	Forschungszentrum Karlsruhe	metadata_generate.xsl		
	metadata_format	text/html	metadata_generate.xsl		
	metadata_type	Text text	metadata_generate.xsl		
Globale Variablen	Name	Inhalt	Gespeichert in Datei ...		
	headline	Textbaustein, der für Webseite angelegt wurde (in <i>textmodules.xml</i>)	XSL-Stylesheet, das <i>headlines_generate.xsl</i> referenziert		
	language	Sprache der Webseite (de/en)	XSL-Stylesheet, das <i>top_link.xsl</i> referenziert		
	overview_path	relativer Dateipfad zu Datei overview_[Projektname]_[Spr.kürzel].xml	XSL-Stylesheet, das <i>graphic_generate.xsl</i> referenziert		
	publishing	Datum der Freigabe	footer_generate.xsl		
	softwarename	- Projektname - leer (bei Webseiten, die projektübergreifend sind)	XSL-Stylesheet, das <i>headlines_generate.xsl</i> referenziert		
	textmodule_path	relativer Dateipfad zur Textmodul-Datei	shared_textmodule_variables.xsl		
	Textbaustein-Variable	siehe Textbausteine	siehe Textbausteine		
Lokale Variablen	Name	Inhalt	Gültig für Funktion ...		
	Suchbegriffe	besteht aus - Textbaustein von Variable <i>keywords_fixed</i> , - Projektname (falls vorhanden), - die vom Autor eingegebenen Schlüsselwörter (Element <i>keywords</i>) - Kontaktpersonen, wenn eine Projektübersichtsseite vorhanden ist - Projektdurchführenden, wenn sie zugestimmt haben (Attribut <i>publish=true</i>)	metadata_generate		
Funktionen	Name	Aufgabe	aufgerufen von...		
	metadata_generate	- verknüpft die einzelnen Inhaltsbausteine von Meta-Tag <i>keywords</i> - weist den Meta-Tags ihre Inhalte zu	XSL-Stylesheet, das <i>metadata_generate.xsl</i> referenziert: - Template <i>root</i>		

Webseitenelement: Metadatenabschnitt						
	Globale Parameter	Name	Inhalt	übergeben von ...	übergeben an ...	
		-	-	-	-	
	Lokale Parameter	Name	Inhalt	übergeben von ...	übergeben an Funktion ...	
		-	-	-	-	
	Links	Linkarten	Link-ID			
		-	-			
	JavaScript Abschnitte	Name	Funktion			
		-	-			
Layout						
	Stylesheet-Name	-				
	Verwendete Klassen	-				
	Verwendete ID	-				
Verwendete Bilder		Bild(-funktion)	Dateiname	Name der globalen Variable		Dateipfad
	-	-	-	-		-

K. Beispiele für XSLT-Stylesheets und JavaScript-Dateien

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="2.0" id="email_link" xpath-default-
namespace="http://fuzzy.fzk.de/~GRID/schema.html" xmlns:ns="
http://fuzzy.fzk.de/~GRID/schema.html" xmlns:fn="http://www.w3.org/2004/07/xpath-functions" exclude-result-prefixes="fn ns
xsl">
3 <xsl:output method="html" encoding="iso-8859-1" />
4
5
6 <!--Globale Variablen*****-->
7 <!--Variablen für feste E-Mail Bilder*****-->
8 <xsl:variable name="at" select="concat(normalize-space($file_path_part),'share/images/symbols/email_at.gif')"/>
9 <xsl:variable name="point" select="concat(normalize-space($file_path_part),'share/images/symbols/email_dot.gif')"/>
10 <xsl:variable name="symbol_email_link" select="concat(normalize-
space($file_path_part),'share/images/symbols/email_link_letter.gif')"/>
11
12 <!--Variablen für Textbaustein-Pfade*****-->
13 <xsl:variable name="tooltip_email_link" select="
docu-
ment($textmodule_path)//type_of_website[@ns:name='shared']/type_of_textmodules[@ns:name='tooltip']/specification_of_ty
pe[@ns:name='email_link']/textmodule[@ns:langua
ge=$language]"/>
14 <!--*****Ende des Stylesheet-Kopfes*****-->
15
16
17 <!--E-Mail Adresse generieren: E-Mail-Text und E-Mail-Link*****-->
18 <xsl:template name="email_generate">
19 <xsl:param name="Email"/>
20 <xsl:param name="style"/>
21 <xsl:message>E-Mail generieren:Text, Link</xsl:message>
22 <xsl:message><xsl:value-of select="$Email"/></xsl:message>
23
24 <!--Brieflogo für E-Mail-Kontakt*****-->
25 <xsl:message>Brieflogo generieren</xsl:message>
26 </img>
27 <xsl:text> </xsl:text>
28
29 <!--E-Mail-Adresse in Textbausteine aufsplitten*****-->
30 <xsl:variable name="email_front_of_dot" select="substring-before($Email, '.')"/>
31 <xsl:message>String vor 1.Punkt: <xsl:value-of select="$email_front_of_dot"/></xsl:message>
32 <xsl:variable name="email_front_of_at" select="substring-before($Email, '@')"/>
33 <xsl:message>String vor @: <xsl:value-of select="$email_front_of_at"/></xsl:message>
34 <xsl:variable name="email_behind_dot" select="substring-after($email_front_of_at,concat($email_front_of_dot, '.'))"/>
35 <xsl:variable name="email_behind_at" select="substring-after($Email, '@')"/>
36 <xsl:variable name="email_front_of_de" select="substring-before($email_behind_at, '.de')"/>
37
38 <!--E-Mail-Adresse verschlüsseln*****-->
39 <xsl:variable name="email_encrypted" select="translate($Email, 'fzk@de.aioubghstmn-', '1234567890.:?!*$/;')"/>
40 <xsl:message>Email encrypted: <xsl:value-of select="$email_encrypted"/></xsl:message>
41
42 <!--E-Mail Text aus Textabschnitten und Bildern zusammensetzen*****-->
43 <a class="{style}" href="javascript:MailTo('{email_encrypted}')" title="{tooltip_email_link}">
44 <xsl:value-of select="$email_front_of_dot"/>
45 </img>
46 <xsl:value-of select="$email_behind_dot"/>
47 </img>
48 <xsl:value-of select="$email_front_of_de"/>
49 </img>
50 <xsl:text>de</xsl:text>
51 </a>
52
53 <xsl:message>Link fertig</xsl:message>

```

```
54 </xsl:template>  
55  
56 </xsl:stylesheet>
```

```
function UnCryptMailTo(mail)
{
    var schluessel="fzk@de.aioubghstmn-";
    var gegenschluessel="1234567890.:?!*$/,";
    var mail_entschluesselt="";

    for (i=0; i < mail.length; i++)
    {
        var gefunden=0;
        for (j=0; j < gegenschluessel.length; j++)
        {
            if (gegenschluessel.charAt(j)==mail.charAt(i))
            {
                mail_entschluesselt=mail_entschluesselt + schluessel.charAt(j);
                var gefunden=1;
            }
        }
        if (gefunden==0)
        {
            mail_entschluesselt=mail_entschluesselt + mail.charAt(i);
        }
    }
    return mail_entschluesselt;
}

function MailTo(mail)
{
    var y=mail;
    //alert (UnCryptMailTo(y));
    location.href = 'mailto:'+UnCryptMailTo(y);
}
```

Erklärung

Ich erkläre hiermit an Eides Statt,

- dass ich die vorliegende Studienarbeit selbstständig angefertigt habe,
- keine anderen als die angegebenen Quellen benutzt habe,
- die wörtlich oder dem Inhalt nach aus fremden Arbeiten entnommenen Stellen, bildlichen Darstellungen und dergleichen als solche genau kenntlich gemacht und
- keine unerlaubte fremde Hilfe in Anspruch genommen habe.

Pfinztal, 10.Oktober 2007

Fabienne Holdermann